On the role of age diversity for effective aging operators

Jansen, Thomas; Zarges, Christine

Published in:
Evolutionary Intelligence

DOI:
10.1007/s12065-011-0051-6

Publication date:
2011

Citation for published version (APA):
Jansen, T., & Zarges, C. (2011). On the role of age diversity for effective aging operators. *Evolutionary Intelligence*, *4*(2), 99-125. https://doi.org/10.1007/s12065-011-0051-6

Noname manuscript No.
(will be inserted by the editor)

On the Role of Age Diversity for Effective Aging Operators

Thomas Jansen · Christine Zarges

Received: date / Accepted: date

Abstract Aging is a general mechanism that some randomized search heuristics employ to increase the diversity of their collection of search points. A more diverse collection of search points is believed to improve the search heuristic's performance for difficult problems. The most prominent randomized search heuristics with aging are evolutionary algorithms and artificial immune systems. While it is known that randomized search heuristics with aging can be very much more efficient than randomized search heuristics without aging the details of the origin of such benefits are difficult to understand. We contribute to this understanding by presenting a detailed and structured analysis of aging. We prove that in addition to diversity with respect to search points diversity with respect to age plays a key role. We analyze different ways of dealing with age diversity by means of theoretical as well as empirical analyses. Major results include a more structured understanding of aging and showcases where age diversity can make the difference between efficient and completely inefficient optimization.

1 Introduction

Randomized search heuristics are a broad class of general search algorithms that comprises nature-inspired heuristics like artificial immune systems [14], evolutionary algorithms [27], simulated annealing [28], and others. Each heuristic implements some general idea of how search should be conducted. These ideas are often borrowed from other fields, evolutionary algorithms mimicking the process of natural evolution, artificial immune systems being modeled after the immune systems of vertebrates, and simulated annealing implementing the process of annealing in metallurgy in an abstract way. Although these randomized search heuristics derive from quite different paradigms they can share some general ideas and exhibit certain similarities. However,

Thomas Jansen
Department of Computer Science, University College Cork, Cork, Ireland
Tel.: +353-(0)21-420-5926
Fax: +353-(0)21-420-5367
E-mail: t.jansen@cs.ucc.ie

Christine Zarges
Lehrstuhl 2, Fakultät für Informatik, TU Dortmund, 44221 Dortmund, Germany

despite these similarities they do differ in the concrete implementations of the concepts and thus, an implementations deriving from one specific paradigm may appear senseless in another one. All such randomized search heuristics are hoped to be efficient on a broad class of problems and are applied when there is only insufficient time or expertise to develop a problem-specific algorithm. While the general idea is to apply a randomized search heuristic 'right out of the box' in practice it is almost always necessary to adjust the randomized search heuristic to the concrete problem at hand to achieve acceptable performance. Thus, in practice, it makes sense to combine ideas from different randomized search heuristics in order to improve the performance of the algorithm.

There are many different ways of tweaking the performance of randomized search heuristics one being the addition of more advanced and sometimes rather complicated mechanisms. One such mechanism is aging where each point in the search space is equipped with an individual age and ages in each round of the search heuristics. A maximal age τ is introduced and each search point with an age exceeding τ is removed from the current collection of search points making room for new and perhaps more promising search points. The mechanism of aging is thought of as increasing the diversity of the collection of search points the randomized search heuristics utilize and is hoped to be helpful for multi-modal problems where simpler search heuristics may get stuck in local optima.

Aging has been used in different kinds of randomized search heuristics, for example in evolutionary algorithms [3,17,19–21,29,31] and artificial immune systems [2,5–13, 32]. It is used for decisions made during the optimization process, e. g., for the selection, for controlling the mutation strength or controlling the size of the collection of search points. Hence, it is not surprising that there exists a large variety of different aging operators.

In this paper, we apply aging during the selection process of a randomized search heuristic. In this context in evolutionary computation aging is often used by assigning age 0 to each new offspring. The age is increased by 1 in each generation. In selection for replacement the age is taken into account: Search points exceeding a pre-defined maximal age τ are removed from the collection of search points. The extreme cases of this aging strategy with $\tau = 0$ and $\tau = \infty$ are known as comma selection and plus selection, respectively [31]. We call this type of aging *evolutionary aging*.

In artificial immune systems a different kind of aging, called *static pure aging* is more common. Again, search points are associated with an individual age and the age is increased by 1 in each round, but in contrast to the former version the offspring inherits by default the age of its parent and is only assigned age 0 if its fitness is strictly larger than its parent's fitness. This aging scheme intends to give an equal opportunity to each improving new search point to effectively explore the landscape. An experimental analysis for this operator was carried out by Castrogiovanni et al. [2]. An elitist version of the aging operator can be obtained by giving the currently best search point in the collection of search points age 0 [13] or, alternatively, by simply forbidding the elimination of the best search point and keeping its age. Moreover, there exists a stochastic version of aging where an search point x with age τ is eliminated with probability $P_{\text{elim}}(\tau) = 1 - e^{-\ln 2/\tau}$ [7–9,13].

Clearly the most important parameter for aging operators is the maximal age τ. It has been shown that the choice of this parameter is both crucial for the performance and difficult to set appropriately [22]. On one hand, the maximal age must not be too small as search points need sufficient time to explore the fitness landscape, where

'sufficient' highly depends on the considered fitness function. On the other hand, the maximal age must not be too large as aging comes into play only when the maximal age is reached. Thus, it needs to be reasonable if aging is to be effective during the optimization process. Clearly, there is no general good setting, the appropriate maximal age depends on the optimization problem [22].

It is known [24] that these different aging strategies have different strengths and weaknesses. While static pure aging can escape from local optima by recognizing stagnation and performing a kind of restart it fails on plateaus, i. e., a set of neighboring points in the search space with equal function value, where it mistakes missing progress in function values for stagnation. On the other hand, evolutionary aging recognizes the random walk on plateaus but fails to escape local optima.

While it is known that employing aging can make the difference between very inefficient and efficient search [22] until very recently in all cases where an artificial immune system with aging was proved to be very much superior to an artificial immune system without aging the same improvement can be achieved when aging is replaced by an appropriate restart strategy [22, 24]. A restart strategy decides at some point of time to stop a randomized search heuristic and start again with a new randomly generated collection of search points. Such restart strategies are conceptually simpler, easier to implement, and computationally cheaper than aging. Thus, it is highly interesting to see what aging can achieve with respect to efficiency of an artificial immune system that cannot be achieved by restarts. Very recently, one example problem has been presented where aging is proven to facilitate a speed-up that cannot be achieved by restarts [25, 26]. This was done by introducing a randomized search heuristic using static pure aging from artificial immune systems and a variation operator known from evolutionary algorithms. Moreover, Jansen and Zarges [25, 26] used a very mild strategy to maintain a certain level of age diversity.

In this paper, we make further investigations with respect to those diversity mechanisms by comparing different selection operators and point out the importance of such diversity strategies. We consider the same example problem and algorithm. We provide upper and lower bounds on its expected optimization time with complete formal proofs. The results are accompanied by an empirical evaluation of the algorithm for different sizes of the collection of search points. Our findings contribute to the theoretical foundation of aging in randomized search heuristics and in particular in artificial immune systems and help to understand the role of different aspects of aging.

In order to make the paper self-contained, we give a detailed and formal description of the complete algorithm and the example problem considered in the following section. In Section 3 we prove upper and lower bounds on the performance of this algorithm on this problem depending on the problem size n, the size μ of the collection of search points, and the maximal age τ. These results hold for a wide range of crossover probabilities p_c. In particular, we point out similarities and differences of the different algorithmic variants. The results are structured in results that hold for all considered variants of static pure aging (Section 3.1), results that depend on the aging strategy but are independent of the replacement strategy (Section 3.2), and results that depend on the concrete instantiation of static pure aging. Since all our theoretical results are asymptotic in nature it makes sense to provide experimental supplements. We do so in Section 4. The empirical findings agree with the proven bounds and provide deeper insights into the different variants and in particular the role of the size of the collection of search points. Finally, we summarize, conclude, and discuss directions for future research in the closing section.

2 The Analytical Testbed

We are interested in the effects age diversity mechanisms can have on the efficiency of the optimization. We analyze this by considering a simple randomized search heuristic as algorithmic framework that is equipped with an aging operator from artificial immune systems and a variation operator from evolutionary algorithms. The search heuristic was already introduced in [25, 26] in order to show that aging can achieve performance improvements that restarts cannot. Moreover, we use the example function from [25, 26] for our considerations.

2.1 The Algorithm

The randomized search heuristic uses a collection of search points of size μ. It works in rounds where in each round all search points grow older, one new search point is generated as random variation of existing search points, its age is decided, search points that are too old are removed and new randomly generated search points are introduced to keep the number of search points constant at μ. A more formal description of the algorithmic framework is given in Algorithm 1.

Algorithm 1 Algorithmic Framework.

1. **Initialization**
   Initialize collection of search points C of size μ.
2. **Aging: Growing older**
   Increase age of all search points in C.
3. **Variation**
   Generate new search point z.
4. **Aging: Age of new search points**
   Decide about the age of the new search point z.
5. **Aging: Removal due to age**
   Remove search points with age exceeding τ.
6. **Selection for Replacement**
   Decide if z is to be inserted in C. Remove or add search points as needed.
7. **Stopping**
   If stopping criterion not met continue at line 2.

We use Algorithm 1 for maximization of an objective function $f\colon \{0,1\}^n \to \mathbb{R}$ and implement the seven modules in very simple ways. The initialization (line 1[1], Algorithm 2) is carried out uniformly at random. All search points are assigned age 0.

Algorithm 2 Initialization.

1. Set $C := \emptyset$. Repeat the following μ times.
2. Select $x \in \{0,1\}^n$ uniformly at random.
3. Set $x.\text{age} := 0$. Set $C := C \cup \{x\}$.

The search points grow older by 1 in each iteration of the main loop (line 2, Algorithm 3).

Variation creates one new search point y by means of k-point crossover and standard bit mutations (line 3, Algorithm 4) known from evolutionary algorithms [27]. The crossover operator is efficient when the collection of search points is sufficiently diverse. Since aging aims at increasing the diversity it is a good idea and interesting test case to

[1] all line numbers from Algorithm 1

combine crossover with aging. In k-point crossover two search points $x, y \in \{0,1\}^n$ are cut into $k+1$ pieces by selecting uniformly at random k *different* cut positions. A new search point is constructed from the pieces by taking all the odd numbered pieces of x (the first, third, ...) and all even numbered pieces of y (the second, fourth, ...) and concatenating them in increasing interleaving order. Usually, k-point crossover with very small values for k is employed, most often $k = 1$ or $k = 2$.

The standard bit mutation operator takes one search point $x \in \{0,1\}^n$ and performs independently for each of the n bits one random experiment. With probability $1/n$ the bit is inverted, otherwise it remains unchanged. We apply these two variation operators in line 3 in the following way. With probability p_c (a parameter of the algorithm), we select two search points from C uniformly at random and perform k-point crossover. The result is subject to mutation. The final result is the new search point z. If no crossover is performed (with probability $1 - p_c$), we select one search point from C uniformly at random and mutate it, the result being the new search point z.

Algorithm 3 Aging: Growing Older

1. For all $x \in C$
2. Set $x.\text{age} := x.\text{age} + 1$.

Algorithm 4 Variation.

1. With probability p_c
2. Select $x, y \in C$ uniformly at random.
   k-Point-Crossover of x and y
3. Select $c_1 \neq c_2 \neq \cdots \neq c_k \in \{0, 1, 2, \ldots, n\}$ uniformly at random.
4. Sort $c_1, \ldots, c_k$ in ascending order. $c_{k+1} := n + 1$.
5. If $c_1 > 0$ Then $h := 1; i := 0$ Else $h := 2; i := 1$.
6. For $j := 1$ To n do
7. If $i = 0$ Then $z[j] := x[j]$ Else $z[j] := y[j]$.
8. If $j \geq c_h$ Then $i := 1 - i$; $h := h + 1$
9. Else
10. Select $x \in C$ uniformly at random.
11. Set $z := x$. Set $y := x$.
12. Independently for each $i \in \{1, 2, \ldots, n\}$
13. With probability $1/n$ set $z[i] := 1 - z[i]$.

As in [26] we consider three different variants to decide about the age of the new search point (line 4, Algorithm 5). The basic idea of static pure aging is to assign age 0 if the new search point is an improvement. Otherwise it inherits its age from the search points it is derived from. As search points created by crossover have two search points as origin, things are less obvious in that case. It is unclear how the comparison with respect to the function value is to be made and what age is to be inherited if no improvement was made. One may believe that these are unimportant details as they only matter in the case of crossover and if the new search point is not good anyway. However, the analytical and experimental results from [26] show, that these details do matter. We discuss the different strategies considered in Section 2.1.1.

Algorithm 5 Outline of Static Pure Aging.

1. If $f(z) > \max\{f(x), f(y)\}$ Then
2. Set $z.\text{age} := 0$.
3. Else
4. Set $z.\text{age} :=$ age of either x or y.

Again following common practice in static pure aging all search points exceeding the maximal age τ are removed and replaced by new random search points to keep the size of the collection of search points constant (line 5, Algorithm 7).

Algorithm 6 Aging: Removal Due to Age.

1. For all $x \in C$
2. If $x.\text{age} > \tau$ Then
3. $C := C \setminus \{x\}$.

The selection for replacement (line 6, Algorithm 7) is the part of the algorithm where age diversity mechanisms come into play. However, the function values are the more important selection criteria. If at least one current search point is removed due to its age the new search point is inserted. Otherwise it is only inserted if its function value is not worse than the worst function value of any of the current search points. If its function value is strictly larger than this value it replaces one current search point that is selected uniformly at random among all search points with worst function value. If its function value is equal to the worst function value, we have to be more careful. If age is considered to be helpful it makes sense to avoid having all current search points of the same age. We discuss different strategies for that in Section 2.1.2.

Algorithm 7 Outline of Selection for Replacement.

1. If $|C| < \mu$ Then
2. If $z.\text{age} \leq \tau$ Then
3. Set $C := C \cup \{z\}$.
4. While $|C| < \mu$ do
5. Select $x \in \{0,1\}^n$ uniformly at random.
6. Set $x.\text{age} := 0$. Set $C := C \cup \{x\}$.
7. Else If $f(z) \geq \min_{x \in C} f(x)$ Then
8. If $f(z) > \min_{x \in C} f(x)$ Then
9. Set $D := \left\{ x \in C \colon f(x) = \min_{x' \in C} f(x') \right\}$.
10. Else
11. Determine D considering an appropriate replacement strategy.
12. Select $y \in D$ uniformly at random.
13. Set $C := (C \cup \{z\}) \setminus \{y\}$.

We avoid to discuss stopping criteria (line 7) by concentrating on the optimization time. Formally, we let the algorithm (Algorithm 1) run forever and consider the first point of time when a global optimum of f is found. As usual we make use of the number of generations as measure of time. Thus, $T_{A,f}$ is the number of generations Algorithm 1 has made when $\max\{f(x) \colon x \in C\} = \max\{f(x) \colon x \in \{0,1\}^n\}$ holds for the first time. Clearly, $T_{A,f}$ is a random variable and we are mostly interested in its mean value $\mathrm{E}\left(T_{A,f}\right)$.

We derive asymptotic results for the optimization time and use the well known Landau notation for describing the asymptotic growth of functions [4] as stated in Definition 1. This implies that our results hold if the parameter n is sufficiently large. Note, that this is different from an analysis that assumes $n \to \infty$.

Definition 1 (Landau notation) Let $f, g \colon \mathbb{N} \to \mathbb{R}$.

- $f(n) = O(g(n)) \Leftrightarrow \exists n_0 \in \mathbb{N}, c \in \mathbb{R}^+ \colon f(n) \leq c \cdot g(n)$, i. e., f does not grow faster than g

- $f(n) = \Omega(g(n)) \Leftrightarrow g(n) = O(f(n))$, i. e., f grows at least as fast as g
- $f(n) = \Theta(g(n)) \Leftrightarrow f(n) = O(g(n))$ and $g(n) = O(f(n))$, i. e., f and g have the same order of growth
- $f(n) = o(g(n)) \Leftrightarrow \lim_{n\to\infty} f(n)/g(n) = 0$, i. e., f grows slower than g
- $f(n) = \omega(g(n)) \Leftrightarrow g(n) = o(f(n))$, i. e., f grows faster than g

We remark that all our results are asymptotic in n (not in any other parameter), i. e., they hold for sufficiently large finite values of n. This also holds if other parameters appear in the stated bounds. Consider for example a bound of $\Theta(\mu \log \mu)$. We get the 'correct' term with respect to the considered asymptotics by replacing the parameter μ with a term depending on n. For $\mu = \log n$ this yields for example $\Theta(\log n \log\log n)$ while constant μ, i. e., $\mu = O(1)$, results in $\Theta(1)$.

Note that the randomized search heuristic we consider has four parameters: the size of the collection of search points $\mu \in \mathbb{N}$, the crossover probability $p_c \in [0, 1]$, the number of crossover points in k-point crossover $k \in \{1, n + 1\}$, and the maximal age $\tau \in \mathbb{N}$. We discuss sensible settings for all four of them.

Since for k-point crossover$(x, x) = x$ holds for all $x \in \{0, 1\}^n$ we see that using crossover requires at least potentially different parents. Thus, the size of the collection of search points $\mu \in \mathbb{N}$ needs to be $\mu \geq 2$. On the other hand, we want to have $\mu = n^{O(1)}$ since otherwise already the initialization requires super-polynomial computational effort and the algorithm cannot be efficient. We investigate the full range of possible sizes of the collection of search points within these limitations.

The crossover probability $p_c \in [0, 1]$ is usually set to some rather large constant like $p_c = 0.8$. Sometimes, very small crossover probabilities are used in proofs (see for example [23]) but in practice this is hardly ever done. We concentrate on crossover probabilities p_c with $\varepsilon \leq p_c \leq 1 - \varepsilon$ for some arbitrarily small positive constant $\varepsilon \in (0, 1)$.

The number of crossover points in k-point crossover is usually a very small constant, $k = 1$ and $k = 2$ are by far the most common choices. We consider $k = O(1)$ here.

It is known that setting the maximal age τ appropriately is extremely difficult and can make the difference between very inefficient and highly efficient [22]. Known results due to Horoba et al. [22] imply some lower bound on the maximal age for the example function considered here. We derive upper bounds on the expected optimization time for all settings of τ that respect this lower bound. Moreover, we derive lower bounds on the expected optimization time for all values of τ. In our experimental evaluation in Section 4 we set τ to an appropriate value.

2.1.1 Variants of Static Pure Aging

We consider all three known variants here and recall their formal definition from [26].

Definition 2 A new search point z that was either created by crossover of x and y or by mutation of x (where we have $x = y$ for notational simplicity) is assigned its age as outlined in Algorithm 5. Line 4 of this algorithm is detailed in three variants as follows.

In *age-based static pure aging* the age is set to the age of the older search point: $z.\text{age} := \max\{x.\text{age}, y.\text{age}\}$.

In *optimistic value-based static pure aging* the age is set to the age of the search point with larger function value, in case of equal function values to the larger age: If $f(x) \neq f(y)$ then $z.\text{age} := \text{argmax}\{f(x), f(y)\}.\text{age}$, else $z.\text{age} := \max\{x.\text{age}, y.\text{age}\}$.

In *pessimistic value-based static pure aging* the age is set to the age of the search point with smaller function value, in case of equal function values to the larger age: If $f(x) \neq f(y)$ then $z.\text{age} := \text{argmin}\{f(x), f(y)\}.\text{age}$, else $z.\text{age} := \max\{x.\text{age}, y.\text{age}\}$.

The idea of static pure aging is to punish a new search point that fails to be an improvement by having it inherit its age. Improvements are rewarded by assigning age 0 and thus a longer lifespan. In the case of crossover the worst punishment possible is to assign the new search point z the larger age of the two other involved search points, x and y. This is what we call age-based static pure aging. This variant has been analyzed in [25,26]. While being simple it does not appear to be entirely fair. The reason the new search point fails to be an improvement could be that a good search point was combined with a bad search point. It therefore makes sense to compare the function values of x and y. If these function values are equal we set the new search point's age to the older age. If, however, the two search points have different function values we have a choice. We can react in an optimistic way to this difference and assign the new search point the age of the better search point. This is what we call optimistic value-based static pure aging. Alternatively, we could be pessimistic and assign the new search point the age of the worse search point. We call this pessimistic value-based static pure aging. The latter two variants have been analyzed in [26].

2.1.2 Variants of Selection for Replacement

Probably the simplest way to maintain a certain degree of age diversity is to replace a search point whose age appears most frequently within the current collection of search points (including the new point itself). This mechanism ensures that the number of different age values among the worst search points does not decrease by exchanging two search points with worst function value. Note that it only affects the worst points in the current selection and only comes into play if another point with this worst function value is inserted.

In [25,26] another mechanism was considered. Here, the new search point replaces one current search point that is selected uniformly at random among all search points with minimal difference in age to the new search point. Again, it is ensured that the number of different age values among the worst search points does not decrease by exchanging two search points with worst function value.

We additionally consider two variants that do not employ age diversity mechanisms. On one hand, we analyze an algorithm that ignores the current age structure and simply replaces one current search point that is selected uniformly at random among all search points with worst function value. Note that this variant corresponds to the standard selection for replacement method in evolutionary algorithms. On the other hand, we consider the extreme case where age diversity is intentionally destroyed by replacing a search point whose ages appears fewest within the current collection of search points (including the new point itself). Similar to the different static pure aging strategies we define a set of replacement strategies formally.

Definition 3 A new search point z that was created during the variation phase of the algorithm replaces a search point from the current collection of search points C as outlined in Algorithm 5. Line 11 of this algorithm is detailed in four variants as follows. In all variants only search points with worst function value are considered for replacement, namely $D' := \{x \in C \colon f(x) = \min_{x' \in C} f(x')\}$.

In *most frequent replacement* the set of search points whose age occurs most frequently within the current selection of search points (including z) is determined. Formally, let $f_a = |\{x \in (C \cup z) \colon x.\text{age} = a\}|$ be the number of occurrences of age a and $f_{\max} = \max_{y \in (C \cup z)} |\{x \in (C \cup z) \colon x.\text{age} = y.\text{age}\}|$ the number of occurrences of the age that occurs most frequently in the current selection of search points. Note, that there may be multiple ages that occur most frequently in C. In this case, all these ages are taken into account. Then, $D = \{x \in C \colon f_{x.\text{age}} = f_{\max}\}$.

In *smallest age distance replacement* a search point from D' with minimal age distance to the new search point is selected uniformly at random, i. e.,

$$D = \left\{ x \in D' \colon |x.\text{age} - z.\text{age}| = \min_{x' \in D'} \left(|x'.\text{age} - z.\text{age}| \right) \right\}.$$

In *random replacement* simply a search point from D' is selected for replacement uniformly at random, i. e., $D = D'$.

In *fewest replacement* the set of search points whose age occurs fewest within the current selection of search points (including z) is determined. Formally, let

$$f_{\min} = \min_{y \in (C \cup z)} |\{x \in (C \cup z) \colon x.\text{age} = y.\text{age}\}|$$

be the number of occurrences of the age that occurs fewest in the current collection of search points. Again, there may be multiple ages that occur fewest in C. Then, $D = \{x \in C \colon f_{x.\text{age}} = f_{\min}\}$.

2.2 The Example Function

For comparison of the three aging variants we consider an example problem where aging provably is essential for being efficient. One such example problem where aging even cannot be replaced by restarts is known [25,26]. We recall its main properties and give a precise definition. Since we want to see beneficial effects due to aging we consider a function with a local optimum that is much easier to find than any global optimum. It is important that a global optimum cannot be found efficiently by means of an appropriate restart mechanism. The considered function $f \colon \{0,1\}^n \to \mathbb{R}$ achieves all this and other goals.

We consider the same problem and compare the different variants described above on it. We start with a formal definition (Definition 4) and add a more detailed informal explanation of the example problem's properties.

Definition 4 The function $f \colon \{0,1\}^n \to \mathbb{R}$ is defined for $n = 4k$, $k \in \mathbb{N}$, and $x \in \{0,1\}^n$ by

$$f(x) = \begin{cases} 2n & \text{if } x = 1^{n/4}0^{n/4}q,\ q \in \{0,1\}^{n/2}, \\ & |q|_1 \geq n/12, \\ n+i & \text{if } x = 1^i0^{n-i},\ i \leq n/4, \\ n - |x|_1 & \text{otherwise.} \end{cases}$$

where $|x|_1 = \textsc{OneMax}(x) = \sum_{i=1}^{n} x[i]$.

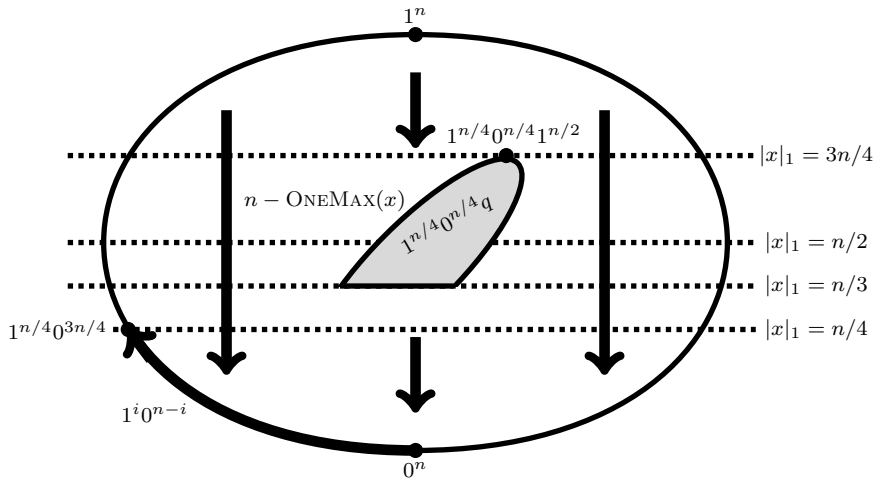


Fig. 1 The example function.

A visualization of f is given in Figure 1 where the considered search space $\{0,1\}^n$ is illustrated as an ellipse. The search point that only contains 1-bits, i. e., 1^n, is located at the top of the ellipse whereas 0^n is at the bottom. Points in between are arranged in a layered fashion and sorted in descending order depending on their number of 1-bits. Important levels are marked with dotted lines. The region of the global optimum is shaded in grey. The bold arrows indicate the direction of increasing fitness of the search points in the rest of the search space.

For the vast majority of the points x in the search space the function value is defined as $n - \textsc{OneMax}(x)$. It is well known [33] that it is easy to follow the direction of increasing function values for such functions. The last point of this type is the all zero bit string 0^n. This is the beginning of a path of Hamming neighbors of the form 1^i0^{n-i} with function values $n+i$ increasing with i. Since it is easy to create the next better point $1^{i+1}0^{n-i-1}$ from 1^i0^{n-i} by means of standard bit mutations (the probability being $(1/n)(1-1/n)^{n-1} \geq 1/(en) = \Theta(1/n)$) we see that the local optimum $1^{n/4}0^{3n/4}$ is easy to find. Points of the form $1^{n/4}0^{n/4}q$ with $q \in \{0,1\}^{n/2}$ and $\textsc{OneMax}(q) \geq n/12$ are special. The set of all these points

$$\text{OPT} := \left\{1^{n/4}0^{n/4}q \colon q \in \{0,1\}^{n/2}, \textsc{OneMax}(q) \geq n/12\right\}$$

equals the set of all global optima of f, i. e.,

$$\text{OPT} = \left\{x \in \{0,1\}^n : f(x) = \max\left\{f(y) \colon y \in \{0,1\}^n\right\}\right\}.$$

The crucial observation is that these points are easy to locate by means of a k-point crossover of the local optimum $1^{n/4}0^{3n/4}$ and some $y \in \{0,1\}^n$ that is chosen uniformly at random but very difficult otherwise. This claim was already proven in [25]. In order to make this article self-contained, we restate the corresponding lemma and argumentation here.

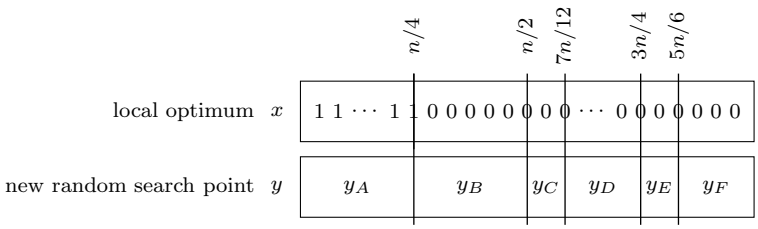


Fig. 2 Visualization of x and y from Lemma 1.

Lemma 1 *Let $x = 1^{n/4}0^{3n/4}$ and $y \in \{0,1\}^n$ be selected uniformly at random. Let OPT be the set of global optima of f. Then, for any $k = O(1)$*

$$\mathit{Prob}\,(k\text{-}\mathit{Point\text{-}Crossover}(x,y) \in \mathit{OPT}) = \Omega(1)$$

holds.

Proof Optimal search points are of the form $1^{n/4}0^{n/4}q$ where $q \in \{0,1\}^{n/2}$ and $\textsc{OneMax}(q) \geq n/12$. We consider one possible way of constructing $z \in \text{OPT}$ by means of k-point crossover of x and y and prove that this happens with a probability that is bounded below by a positive constant. Note that we do not aim at deriving tight bounds on this probability and sacrifice pointless accuracy in favor of simplicity of proof.

Consider a crossover point c with $c \in \{0,1,\ldots,n\}$ selected uniformly at random. For any constants $0 \leq \delta < \delta' \leq 1$ we have that $\delta n \leq c < \delta' n$ holds with probability at least $\varepsilon > 0$ where ε is a positive constant depending on $\delta' - \delta$.

Consider crossover points $c_1 < c_2 < \cdots < c_k \in \{0,\,1,\,\ldots,\,n\}$ (with $k \in \mathbb{N}$ and $k = O(1)$) selected uniformly at random. We have $(1/2)n \leq c_1 < (7/12)n$ ($c_1 \in y_C$), $(3/4)n \leq c_2 < (5/6)n$ ($c_2 \in y_E$), and $(5/6)n \leq c_i \leq n$ ($c_i \in y_F$) simultaneously for all $i \in \{3,4,\ldots,k\}$ with probability at least ε^k where $\varepsilon > 0$ is some constant. Note that this holds for any constant k (and that for very small values of k like $k = 1$ the conditions on c_2 and c_i with $i > 2$ are empty and thus trivially hold).

In this situation the crossover of $x = 1^{n/4}0^{n/4}$ and y (where $y \in \{0,1\}^n$ uniformly at random) is carried out as can be seen in Figure 2. Having $(1/2)n \leq c_1 < (7/12)n$ implies that the leftmost $n/2$ bits of z equal $1^{n/4}0^{n/4}$ since these bits are copied from x. The bits between c_1 and c_2 ($c_2 = n$ in the case of 1-point crossover) are copied from y. Since we have $(1/2)n \leq c_1 < (7/12)n$ and $(3/4)n \leq c_2 < (5/6)n$ we know that these are at least $(3/4)n - (7/12)n = (1/6)n$ bits copied from y. Clearly, these bits are distributed uniformly at random. The expected number of 1-bits in these $(1/6)n$ bits equals $(1/12)n$ and we have at least $(1/12)n$ 1-bits among these bits with probability at least $1/2$. Thus, we have $\text{Prob}\,(z \in \text{OPT}) \geq (1/2) \cdot \varepsilon^k = \Omega(1)$ as claimed. □

It is easy to see that the global optima of f are difficult to find in a different way. For all points $x \in \text{OPT}$ we have $n/3 \leq \textsc{OneMax}(x) \leq (3/4)n$ and thus there are always exponentially many points with the same number of 1-bits. For each number i of 1-bits let OPT_i denote the set of bit strings from OPT with this number of 1-bits, i. e., $\text{OPT}_i = \{x \in \text{OPT} \colon \textsc{OneMax}(x) = i\}$. Let $\overline{\text{OPT}_i} = \{x \in \{0,1\}^n \colon \textsc{OneMax}(x) = i\} \setminus \text{OPT}_i$ denote the other strings with the same number of 1-bits. Clearly, we have $|\text{OPT}_i| \,/\, \left|\overline{\text{OPT}_i}\right| = 2^{-\Omega(n)}$ and we conclude that it is highly unlikely to find OPT by pure random sampling. This implies that restarts do not help. Also randomized search heuristics that are efficient on OneMax are unlikely to encounter OPT since

they quickly leave the part of the search space with these numbers of 1-bits. Thus, they sample only a polynomial number of such bit strings and encounter OPT only with probability $n^{O(1)} \cdot 2^{-\Omega(n)} = 2^{-\Omega(n)}$.

Note that we do not claim that no search heuristic without aging can be efficient on f. Clearly, search heuristics choosing some $x \in \text{OPT}$ as initial search point optimize f with a single function evaluation. While such search heuristics obviously cheat on f by incorporating too much knowledge about it into their 'search strategy' there are other mechanisms that 'cheat' in less obvious ways. Mechanisms that maintain a high degree of diversity in the collection of search points are another way of coping with multimodal problems. Friedrich et al. [16] consider several such mechanisms, among those one that preserves diversity on the level of fitness values (that the authors incorrectly denote as phenotypic diversity). This mechanism works well on many functions where the number of function values is small, i. e., polynomially bounded. Clearly, for noisy functions or continuous functions in $\mathbb{R}^n$ such a mechanism cannot work. Moreover, for Ackley's well-known trap function [1] it achieves highly efficient optimization, a clear indication that this mechanism is cheating in some way. When discussing example functions it is completely pointless to discover or, even worse, invent search heuristics that are efficient on the example function under consideration. The point of considering example functions is to exhibit situations that highlight the usual working patterns of commonly used randomized search heuristics. While randomized search heuristic are very often used in practice and aging is a commonly used mechanism to improve their performance on difficult problems this fitness-based diversity mechanism is not commonly used. We therefore claim that reasonable search heuristics without aging and crossover fail to be efficient on this example function f.

We have seen that appropriate k-point crossover operations can yield a global optimum of f with good probability. For the smallest age distance replacement it was shown in [25] that the probability that the randomized search heuristic under consideration (Algorithm 1) performs such crossover operations is sufficiently large to be efficient on f. Moreover, it was shown that the algorithm is efficient for a very large number of settings of its parameters and thus this good performance is achieved in a robust and reliable way. In the next section we extend our analysis to the different variants of selection for replacement described in Definition 3. For the sake of completeness, the exact results for smallest age distance from [25] are also repeated.

The example function f as defined in Definition 4 is very specific. It is used as a vehicle to demonstrate and formally prove a number of properties of the aging operators we consider here. The same effects can be observed when optimizing other problems, too. For example, it is not essential that f contains exactly one local optimum where a crossover with random search points is needed to locate the global optimum. Functions with several of such local optima would not be very much different. However, it is essential that a partial restart is needed for the optimization. If a complete restarts suffices aging is not needed and may be replaced by an appropriate restart strategy.

3 Theoretical Analyses

In this section we prove upper and lower bounds for Algorithm 1 using the different strategies for static pure aging and selection for replacement from Definition 2 and Definition 3. We see that the algorithm is efficient on our example problem f (Definition 4) if most frequent replacement or smallest age distance replacement are used

together with an arbitrary static pure aging variant given that its parameters are chosen appropriately. Moreover, we will show that the strategies random replacement and fewest replacement lead to inefficient optimization. The different algorithmic variants we consider share many properties, in particular while approaching the local optimum. We reflect this by starting with a section on these common properties.

3.1 Common Properties of All Considered Aging Variants

The most critical parameter is the maximal age τ employed in aging. We use a common lower bound $\tau = \omega(\mu n \log \mu)$ for all upper bounds on the expected optimization time. The work of Horoba et al. [22] indicates that this bound is sufficiently large for optimizing f. Apart from this the algorithm as well as our proofs work for most settings of the other parameters. We require $\mu \geq 2$ since we need crossover and the usual bound $\mu = n^{O(1)}$, thus the size of the collection of search points μ is almost completely unrestricted. The crossover probability p_c can be set almost arbitrarily. We deal with any value $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for some arbitrarily small constant $\varepsilon > 0$. Setting p_c in this way the concrete value of p_c has no influence on the asymptotic expected optimization time. Note, however, that having p_c converge to either 0 or 1 may change things considerably. While it is not difficult to adjust our upper bound to such settings we refrain from considering these rather unusual cases. As already pointed out in Lemma 1 the number k of crossover points used in k-point crossover is not very important as long as it is bounded above by a constant. Clearly, smaller values are better for f and we restrict our attention to the commonly used 1-point crossover in Section 4 when performing experiments.

The different variants of Algorithm 1 under consideration behave very similarly until the local optimum is reached for the first time. The main difference of the considered variants is the way a global optimum can be constructed by recombination of a local optimum and a randomly chosen search point like in Lemma 1. This can happen when we have at least two search points in the local optimum and some but not all of those locally optimal search points are removed due to their age. We call such an event a partial restart. It is unclear how likely it is that such a partial restart occurs. Moreover, we need to derive the probability that given that a partial restart occurs an appropriate crossover operation is executed afterwards. We leave these two questions open for the moment and first prove parameterized lower and upper bounds for all strategies. In these bounds the probabilities for these two events appear as unknowns. Afterwards, we investigate them separately for the different replacement strategies. We start with the upper bound on the optimization time of the Algorithm 1 and the parameter settings discussed above.

Lemma 2 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using an arbitrary strategy for static pure aging from Definition 2 and an arbitrary strategy for selection for replacement from Definition 3, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Moreover, let p denote the probability that a partial restart occurs and q the probability for an appropriate crossover operation creating the global optimum after such partial restart.

Then, $E\left(T_{A,f}\right) = O\left(p^{-1}q^{-1}\left(\tau + n^2 + \mu n \log n\right)\right)$.

The probabilities p and q depend on n, μ, p_c, k and τ and the aging and replacement strategies used. For the sake of readability we just write p and q since the parameters are obvious from the context.

Proof (of Lemma 2) There are three regions of the search space that correspond to phases of a run of Algorithm 1 on f. In the vast majority of the search space the fitness value is given as $n - \textsc{OneMax}(x)$. Due to our lower bound on the maximal age τ this part can be optimized as Algorithm 1 without aging, i. e. the so-called (μ+1) EA, optimizes OneMax. The additional use of crossover cannot increase the asymptotic growth of the expected optimization time here since with probability $1-p_c \geq \varepsilon = \Omega(1)$ no crossover is performed. Thus the expected optimization of $O(\mu n + n \log n)$ [33] carries over. Second, there are the bit strings of the form $1^i 0^{n-i}$ with $i \leq n/4$. Again, due to our lower bound on the maximal age τ this part can be optimized as the (μ+1) EA optimizes LeadingOnes, a well-known example function that is given by $\textsc{LeadingOnes}(x) = \sum_{i=1}^{n} \prod_{j=1}^{i} x[j]$. It is known [33] that the (μ+1) EA's expected optimization time is $O\big(n^2 + \mu n \log n\big)$ and this bound carries over, too.

Finally, we are interested in constructing a global optimum by recombination of a local optimum and a randomly chosen search point like in Lemma 1. This happens with probability q after a partial restart that in turn occurs with probability p. Both experiments follow the geometric distribution. Thus, after expected q^{-1} partial restarts a globally optimal search point is created. Moreover, Algorithm 1 requires expected p^{-1} trials to perform a partial restart.

Clearly, the time we need to wait until a search point is removed due to its age (or earlier due to other reasons) is at most τ which concludes the proof. □

The following lower bound on the optimization time of the Algorithm 1 holds for all possible values of τ and the same settings of μ and p_c as before.

Lemma 3 *Consider $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using an arbitrary strategy for static pure aging from Definition 2 and an arbitrary strategy for selection for replacement from Definition 3, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1-\varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and any maximal age τ (with $\tau = 2^{O(n)}$).*

Moreover, let p denote the probability that a partial restart occurs and q the probability for an appropriate crossover operation creating the global optimum after such partial restart.

Then, $E\left(T_{A,f}\right) = \Omega\big(p^{-1}q^{-1}\left(\tau + n^2 + \mu n \log n\right)\big)$.

Proof As in the proof of Lemma 2 there are three regions in the search space that correspond to phases of a run of Algorithm 1 on f: the part where the fitness value is given by $n - \textsc{OneMax}(x)$, the LeadingOnes path to the local optimum and the region of the global optimum.

First assume that the maximal age τ is sufficiently large, say $\omega(\mu n \log \mu)$. In this case a lower bound can be proven similarly to the (μ+1) EA on LeadingOnes [33]. For this we need to show that the LeadingOnes-like path is first reached by a search point with a number of 0-bits that is $\Omega(n)$. We pick $(7/8)n$ here somewhat arbitrarily.

As already discussed in Section 2 the probability that the algorithm initializes in some $x \in \text{OPT}$ is $2^{-\Omega(n)}$. Moreover, the probability to encounter OPT by optimizing

the $n - \textsc{OneMax}(x)$ part in the first phase is $2^{-\Omega(n)}$. Analogously we can show that Algorithm 1 first hits the path to the local optimum with a search point with at most $n/8$ 1-bits with probability $1 - 2^{-\Omega(n)}$. Let L_i denote the set of bit strings with i 1-bits. Then, the local optimum belongs to $L_{n/4}$ and the probability that Algorithm 1 reaches the path at some point with at least $n/8$ 1-bits is

$$(n/8)/\sum_{i=1}^{n/8} |L_i| = 2^{-\Omega(n)}.$$

Note that crossover does not increase the probability of finding the path with a larger number of 1-bits as after initialization all search points have at least $n/16$ 0-bits within the first $n/4$ of the bit string with probability $2^{-\Omega(n)}$ and the number of 0-bits is increasing during the $n - \textsc{OneMax}(x)$ phase.

We now investigate the probability to make some progress on the path. Assume, there are b search points on the path. Then, the probability to create a new best search point on the path by means of mutation is $O(b/(\mu n))$ as one of the b search points has to be selected and at least a mutation of a single bit is needed.

Considering a crossover operation of a search point $x = 1^i 0^{n-i}$ on the path and a search point y that has not yet reached the path, we easily get the same upper bound. In order to create another point on the path, x has to be selected as first parent which happens with probability at most b/μ. Moreover, $y_{i+1} = 1$ is needed to increase the number of leading ones and thus, yield progress on the path. This event has probability at most $1/2$. Finally, we require $c_1 = i$ for the first crossover point in order to copy the old i leading ones from x and the additional one from y. This happens with probability at most $1/n$.

Clearly, it is not possible to create a new best search point with crossover of two search points on the path. Points on the path have the form $1^i 0^{n-i}$ and if the crossover of $1^i 0^{n-i}$ and $1^j 0^{n-j}$ yields another path point then this path point is $1^k 0^{n-k}$ with $\min\{i, j\} \leq k \leq \max\{i, j\}$. Now consider a crossover where a search point participates that is not a path point. We consider the sequence of its ancestors. If any of these is a path point we call this point a former path point. For former path points the same reasoning implies as for path points. Repairing them back into path points cannot increase the probability for progress on the path. If the point is not a former path point it was created by random initialization and optimization of $n - \textsc{OneMax}(x)$. The local optimum is $1^{n/4}0^{3n/4}$. The probability that a purely random search point also starts with a sequence of $n/4$ 1-bits equals $2^{-n/4}$. When optimizing $n - \textsc{OneMax}(x)$ this probability decreases since the number of 1-bits cannot increase. Thus, with probability very close to 1 crossover does not help in finding the local optimum. Thus, the first search point in the local optimum has to be created by means of mutation and thus, gets age 0.

We still need to consider if crossover can asymptotically decrease the time we need to increase the number of best search points on the path from 1 to b. The probability to increase this number from b to $b + 1$ by means of mutation is $\Theta(b/\mu)$ since we need to select one of the b best search points and do not flip any bits during mutation which happens with probability $1/4 \leq (1-1/n)^n \leq 1/e$. For crossover it is necessary to select a currently best search point as first parent and thus, the probability to create a copy by means of crossover is also $O(b/\mu)$.

Altogether, we see that the lower bound for the LeadingOnes part carries over from the (μ+1) EA [33] and we get $\Omega\big(n^2 + \mu n \log n\big)$ for the second phase. Note, that this dominates the upper bound for the first phase.

Once the complete collection of search points is on the path to the local optimum or in the local optimum, i. e., of the form 1^i0^{n-i} for possibly different values of i with $1 \leq i \leq n/4$ for all of them, the global optimum can only be reached via a direct mutation to the OPT region. Such a mutation has probability at most

$$\binom{n}{n/12} \cdot \left(\frac{1}{n}\right)^{n/12} \leq \frac{1}{(n/12)!}$$

as at least $n/12$ bits in the second half of the bit string have to flip. Thus, the probability to create a global optimum by means of mutation is $n^{-\Omega(n)}$. Clearly, it is not possible to create a global optimum with crossover of two search points on the path.

As the waiting time for a partial or complete restart is at least τ, we need time $\Omega\big(\tau + n^2 + \mu n \log n\big)$ to get into a situation where the first search point in the local optimum is removed due to the maximal age. As in Lemma 2 we need expected q^{-1} partial restarts to create a globally optimal search point and expected p^{-1} trials to perform a partial restarts which proves the claimed lower bound in the case where the maximal age τ is sufficiently large.

If the maximal age τ is not sufficiently large the search process is slowed down as it becomes harder to reach the local optimum. If τ is very small almost constantly new search points are created uniformly at random. In t time steps, at most μ/τ new search points are created in this way. Each of these new search points is equal to a specific globally optimal search point with probability 2^{-n} as discussed above. Such a process finds some of the less than $2^{n/2}$ global optima in an expected number of more than $2^{n/2}$ steps. □

3.2 Properties of Static Pure Aging Independent of Replacement Strategies

Before considering the different replacement strategies, we further discuss the concept of partial restarts. Remember that *partial restart* denotes the event when we have at least two search points in the local optimum and some but not all of those locally optimal search points are removed due to their age. A necessary condition for such an event is that by the time when the maximal age τ is reached by one of the search points in the local optimum, at least two search points with different ages are in the local optimum. Moreover, there is no possibility to create another locally optimal search point with different age once all search points have reached the local optimum as descendants always inherit the age of one of their parents. Thus, the two search points in the local optimum with different ages have to be created while the collection of search points approaches the local optimum. Additionally, after all search points have reached the local optimum, this property of the age structure within the collection of search points has to be preserved by the replacement strategy until the maximal age τ is reached by one of the search points in the local optimum.

The first condition, i. e., creating two search points with two different ages in the local optimum, is independent of the replacement strategy. We therefore analyze the probability for this event and the three static pure aging variants before looking closer at the different replacement strategies.

Lemma 4 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4 and Algorithm 1 using age-based or pessimistic value-based static pure aging from Definition 2 and an arbitrary strategy for selection for replacement from Definition 3, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Let p_1 be the probability for the event that two search points with different ages enter the local optimum before all search points are in the local optimum. Then, $p_1 = \Omega(1)$.

Proof Consider a point of time when $\max\{f(x)\colon x \in C\}$ is increased to $(5/4)n$, i. e., a first locally optimal search point x is produced. Clearly, this search point enters the collection of search points and is assigned age 0. Note, that at this point of time all other search points have age different from x. At later points of time descendants of x may have the same age.

We claim that at this point of time all other members of the collection of search points also have form 1^i0^{n-i} for different values of i with $i < n/4$ with probability close to 1. In the proof of Lemma 3 we saw that with probability close to 1 the first individual to enter the global optimum does so from the path and needed $\Omega\big(n^2 + \mu n \log n\big)$ steps to get there. Since all points on the path have larger function value than the other search points the probability to increase the number of search points on the path from some value v to $v+1$ is $\Omega(v/\mu)$: it suffices to select one such individual (probability v/μ) and do not change it via mutation (probability $(1-p_c)(1-1/n)^n = \Omega(1)$). Thus, the process of getting the whole collection of search points on the path has expected length $O(\mu \log \mu)$ (similar to the coupon collector process [30]), much smaller than the $\Omega\big(n^2 + \mu n \log n\big)$ steps needed in expectation to reach the local optimum. This implies the claim.

We consider the following $\tau = \omega(\mu n \log \mu)$ generations. We prove that within these τ steps with probability $p = \Omega(1)$ another locally optimal search point with age different from x.age enters the collection of search points. To this end, we consider crossover.

Consider $x = 1^{n/4}0^{3n/4}$ and $y = 1^i0^{n-i}$ with $0 \leq i < n/4$. We have

$$\text{Prob}\left(k\text{-Point-Crossover}(x,y) = x\right) = \Omega(1)$$

in the same way as we obtained Lemma 1. Note that the offspring has the same fitness as x. Thus, in the pessimistic value-based variant, the offspring's age is set to the age of y which is different to the age of x, proving $p_1 = \Omega(1)$ in that case.

For the age-based variant we need to be slightly more careful since the offspring gets initial age $\max\{x.\text{age}, y.\text{age}\}$ and hence, it is not clear whether x is older than y or vice versa. When x entered the collection of search points it was the search point with minimal age 0. Thus, all other search points have age different from x unless they are created as descendants of x. We now consider the following $\Theta(\mu)$ steps. Clearly, in these steps this first search point x or one of its copies is selected for reproduction involving crossover with probability $\Omega(1)$. The expected number of descendants of x made in these $\Theta(\mu)$ steps is bounded above by $O(1)$ with probability $1-\delta$ for any constant $\delta > 0$. The number of search points in the collection of search points that may have improved within these steps is bounded by $O(\mu/n)$ since improvements can only occur via mutations but not by crossover alone as seen in the proof of Lemma 3. Thus, we only have $O(\mu/n)$ improved search points within these steps and this also holds with probability $1-\delta$ for any constant $\delta > 0$. We conclude that there are $\Omega(\mu)$ search points on the path with age larger than x. Thus, one of these is selected together with x with

probability $\Omega(1)$ in this $\Theta(\mu)$ steps we consider. These two parents produce another locally optimal offspring that will have age different from x with probability $\Omega(1)$. □

For the considered problem, the only difference between aging in the optimistic and pessimistic value-based variant is the way partial restarts can be achieved, i. e., the way a second age can enter the local optimum. The main difference is that crossover no longer helps in creating another locally optimal search point with age different from the first search point entering the local optimum. If we perform crossover of $x = 1^{n/4}0^{3n/4}$ and $y = 1^i0^{n-i}$ with $i < n/4$ the age of the new search point is given by the age of the better search point, i. e., by x.age. This is no different from a copy of x. Thus, we need to rely on mutations only.

Lemma 5 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4 and Algorithm 1 using optimistic value-based static pure aging from Definition 2 and an arbitrary strategy for selection for replacement from Definition 3, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1-\varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Let p_1 be the probability for the event that two search points with different ages enter the local optimum before the whole collection of search points has reached the local optimum. Then,

$$p_1 = \begin{cases} \Theta(\mu \log(\mu)/n) & \text{if } \mu \leq \delta \cdot n/\log n \text{ for constant } \delta > 0 \text{ sufficiently small} \\ \Theta(1) & \text{otherwise} \end{cases}.$$

Proof Consider the first search point x that enters the local optimum.

We show that by the time half of the collection of search points is taken over by copies of x the rest of the collection of search points are all of the form $1^{(n/4)-1}0^{3(n/4)+1}$ with probability close to 1. If there are b copies of x the probability to increase the number of copies to $b+1$ is $O(b/\mu)$. Since initially we have $b = 1$ we obtain $\Omega(\mu \log \mu)$ as lower bound for creating $\mu/2$ copies of x. On average in these steps already copies of the second best have been produced. Since the selection for reproduction is uniform these copies are selected with higher probability than the first single best. This yields that all worse search points will be removed.

If there are b copies of a second best search point the probability to create a better search point is $O(b/(\mu n))$ since one of them has to be selected and at least a mutation of a single bit is needed. However, in the situation described above and as long as $b = \Theta(\mu)$, the probability to create another locally optimal search point via mutation is $\Omega(1/n)$. The expected time for increasing the number of copies from $b = \mu/2$ to $b = c\mu$ for some constant $c > 1/2$ is also $\Theta(\mu \log \mu)$. Again, this holds due to the similarity to to the coupon collector process [30]. Hence, the probability to create another locally optimal search point with age different from x is $\Omega(1/n)$ for $\Theta(\mu \log \mu)$ steps. After this number of steps this probability can decrease even to 0 since after that time the whole collection of search points may be in the local optimum. In the following we use c/n (for some sufficiently small positive constant c) as lower bound on this probability.

We start with the lower bound on the probability p_1. Assume $\mu \leq n/(cc' \log n)$, i. e., $\delta \leq 1/(cc')$, for positive constants c and c'. Using $(1-x)^y \leq e^{-xy}$ in $(*)$ and $e^x \leq 1/(1-x)$ for $x < 1$ in $(**)$, the probability that another locally optimal search

point with age different from x is created within these $\Theta(\mu \log \mu)$ steps is

$$p_1 \geq 1 - \left(1 - \frac{c}{n}\right)^{c' \mu \log \mu} \overset{(*)}{\geq} 1 - e^{-cc' \frac{\mu \log \mu}{n}} \overset{(**)}{\geq} 1 - \frac{1}{1 + cc' \mu \log(\mu)/n}$$
$$= 1 - \frac{n}{n + cc' \mu \log \mu} = \frac{cc' \mu \log \mu}{n + cc' \mu \log \mu} \geq \frac{cc' \mu \log \mu}{2n}.$$

Otherwise we get

$$p_1 \geq 1 - \left(1 - \frac{c}{n}\right)^{c' \mu \log \mu} \geq 1 - e^{-cc' \frac{\mu \log \mu}{n}} = 1 - e^{-\Omega(1)} = \Omega(1).$$

Since p_1 is a probability (and thus $p_1 \leq 1$), $p_1 = \Omega(1)$ implies $p_1 = \Theta(1)$.

We still need to consider the upper bound for $\mu \leq \delta n / \log n$. The probability to create another locally optimal search points is at most $1/n$ as at least a mutation of a single bit is needed. Again, assume $\mu < n/\log n$. Then, $\mu \log \mu/(n-1) \leq 1$ holds. Using $1 - e^{-x} \leq 2x/(1+2x)$ for $x \leq 1$ in $(***)$ we see analogously to the calculations above that another locally optimal search point with different age is created in $\Theta(\mu \log \mu)$ steps with probability

$$p_1 \leq 1 - \left(1 - \frac{1}{n}\right)^{c\mu \log \mu} \leq 1 - e^{-\frac{c\mu \log \mu}{n-1}}$$
$$\overset{(***)}{\leq} \frac{2c\mu \log(\mu)/(n-1)}{1 + 2c\mu \log(\mu)/(n-1)} = \frac{2c\mu \log \mu}{n - 1 + 2c\mu \log \mu} \leq \frac{2c\mu \log \mu}{n}$$

for some positive constant c, concluding the proof of the lemma. □

We are now ready to consider the different replacement variants.

3.3 Smallest Age Distance Replacement Strategy

The smallest age distance replacement was already analyzed in [25,26]. For the sake of completeness we restate these results here.

Theorem 1 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using smallest age distance replacement from Definition 3. a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Then, we have

$$E\left(T_{A,f}\right) = O\Big(\mu \cdot \Big(\tau + n^2 + \mu n \log n\Big)\Big)$$

for the age-based and pessimistic value-based static pure aging and

$$E\left(T_{A,f}\right) = O\left(\left(\mu + \frac{n}{\log \mu}\right) \cdot \Big(\tau + n^2 + \mu n \log n\Big)\right)$$

for the optimistic value-based static pure aging.

Proof From Lemma 2 we know that $\mathrm{E}\left(T_{A,f}\right) = O\big(p^{-1}q^{-1}\left(\tau + n^2 + \mu n \log n\right)\big)$ holds where p is the probability that a partial restart occurs and q the probability for an appropriate crossover operation creating the global optimum after such partial restart. For the age-based and pessimistic value-based variant we see that it suffices to prove that $p^{-1}q^{-1} = O(\mu)$ holds, whereas for the optimistic value-based variant we need to show $p^{-1}q^{-1} = O(\mu + n/\log \mu)$.

Due to Lemma 4 the probability to have two locally optimal search points with different age when all search points have reached the local optimum is $p_1 = \Omega(1)$ for the first two variants and thus $p^{-1} = O(1)$. Due to Lemma 5 we have $p_1 = \Theta(\mu \log(\mu)/n)$ for sufficiently small $\mu = O(n/\log n)$ and $p_1 = \Theta(1)$ otherwise for the latter variant, leading to $p^{-1} = O(1 + n/\mu \log \mu)$. Note that once we have at least two search points that are both locally optimal but have different age in the collection of search points this will always be the case until a restart happens. This is due to the smallest distance replacement where in case of equal fitness an search point with minimal age difference in selected for replacement. Hence, $p = p_1$ follows.

We still need to derive the probability q for the different variants. Consider the point of time when the age of x, the first search point that has reached the local optimum, exceeds τ. In this generation x and all its copies with identical age are removed and replaced by purely random search points. The expected takeover time for x to take over the complete collection of search points is $O(\mu \log \mu)$. If there are other points in the local optimum (with age different from x) the time until all search points are locally optimal can only be smaller. Since $\tau = \omega(\mu n \log n)$ holds we have with probability close to 1 that all other search points are also locally optimal. Thus, after removing b copies of x we have a collection of search points with $\mu - b$ local optima and b random search points. Remember that $0 < b < \mu$ holds since we have a partial restart. Thus, with probability

$$q = \Omega\left(\frac{b}{\mu} \cdot \frac{\mu - b}{\mu}\right) = \Omega\left(\frac{1}{\mu}\right)$$

the global optimum is produced as next offspring. This establishes that on average $q^{-1} = O(\mu)$ such partial restarts suffice. Putting these things together, we get the claimed upper bound. □

Theorem 2 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using smallest age distance replacement from Definition 3. a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and any maximal age τ (with $\tau = 2^{O(n)}$).*

Then, we have

$$E\left(T_{A,f}\right) = \Omega\Big(\tau + n^2 + \mu n \log n\Big)$$

for the age-based and pessimistic value-based static pure aging and

$$E\left(T_{A,f}\right) = \Omega\left(\left(1 + \frac{n}{\mu \log \mu}\right) \cdot \Big(\tau + n^2 + \mu n \log n\Big)\right)$$

for the optimistic value-based static pure aging.

Proof From Lemma 3 we know that $\mathrm{E}\left(T_{A,f}\right) = \Omega\big(p^{-1}q^{-1}\left(\tau + n^2 + \mu n \log n\right)\big)$ holds where p is the probability that a partial restart occurs and q the probability for an appropriate crossover operation creating the global optimum after such partial restart.

Clearly, we need at least one successful partial restart to obtain the global optimum. Thus, we have the trivial lower bound for the age-based and pessimistic value-based variant following directly from Lemma 3. For the optimistic value-based variant we additionally know $p^{-1} = \Omega(1 + n/\mu \log \mu)$ due to Lemma 5, concluding the proof. □

We see that in the case of smallest age distance replacement the gap between the lower and the upper bound on the expected optimization time is $\Theta(\mu)$. This stems from the fact that we bounded the probability q for creating a globally optimal search points by means of crossover after a partial restart by $q = \Omega(1/\mu)$ and $q = O(1)$ respectively.

The smallest distance replacement has the property that the initial age structure of the collection of search points in the local optimum is not changed after the last search point enters the local optimum. This is due to the fact that after that point of time no new age value can enter the collection of search points as in the case of an non-improving iteration the age is always inherited of one of the parents. Hence, there is always at least one search point in the collection of search points that has the same age as the new search point. Since the age distance to these points is zero, simply two search points with the same age are exchanged. This is different for the most frequent replacement as shown in the next subsection.

3.4 Most Frequent Replacement Strategy

The most frequent replacement is probably the easiest and most direct way of preserving some degree of diversity with respect to age. Like smallest distance replacement it is effective enough to yield efficient optimization since again once we have at least two search points that are both locally optimal but have different age in the collection of search points this will always be the case until some restarts happens. Thus, the upper and lower bounds for smallest age distance replacement simply carry over to the most frequent replacement strategy as stated in the following two corollaries.

Corollary 1 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using most frequent replacement from Definition 3. a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Then, we have

$$E\left(T_{A,f}\right) = O\Big(\mu \cdot \Big(\tau + n^2 + \mu n \log n\Big)\Big)$$

for the age-based and pessimistic value-based static pure aging and

$$E\left(T_{A,f}\right) = O\left(\left(\mu + \frac{n}{\log \mu}\right) \cdot \Big(\tau + n^2 + \mu n \log n\Big)\right)$$

for the optimistic value-based static pure aging.

Corollary 2 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using most frequent replacement from Definition 3, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and any maximal age τ (with $\tau = 2^{O(n)}$).*

Then, we have

$$E\left(T_{A,f}\right) = \Omega\Big(\tau + n^2 + \mu n \log n\Big)$$

for the age-based and pessimistic value-based static pure aging and

$$E\left(T_{A,f}\right) = \Omega\left(\left(1 + \frac{n}{\mu \log \mu}\right) \cdot \Big(\tau + n^2 + \mu n \log n\Big)\right)$$

for the optimistic value-based static pure aging.

In contrast to smallest age distance replacement most frequent replacement changes the initial distribution of the different age values. It aims at obtaining and preserving a completely balanced distribution of ages within the collection of search points. Given such a balanced distribution better bounds on the expected optimization time can be proved. We consider Lemma 2 and remember that p denotes the probability for a partial restart and q denotes the probability that this partial restart is successful, i. e., generates a globally optimal search point. Now we replace p and q by p' and q' where p' denotes the probability to have a collection of search points completely within the local optimum with $r+1$ different ages. In this situation we will have r partial restarts within the next τ steps or the global optimum is found. If q' denotes the probability that at least one of these partial restarts leads to the global optimum we obtain essentially the same bound as in Lemma 2. We state this as a corollary.

Corollary 3 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using an arbitrary strategy for static pure aging from Definition 2 and an arbitrary strategy for selection for replacement from Definition 3, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Moreover, let p' denote the probability that within the next τ steps r partial restarts occur and q' the probability for an appropriate crossover operation creating the global optimum after one of these partial restarts.

Then, $E\left(T_{A,f}\right) = O\big((p' \cdot q')^{-1}\left(\tau + n^2 + \mu n \log n\right)\big)$.

Now we consider the different static pure aging strategies from Definition 2. The main difference between optimistic value-based aging and the two other variants (pessimistic value-based aging and age-based aging) is that in optimistic value-based aging a new age can only be introduced to the local optimum via mutation. Crossover operations without mutation need to involve one search point that already is a local optimum and one other search point. This other search point is worse with respect to function value in comparison to the other search point. Thus, in the pessimistic value-based variant this age is used and introduced as a (potentially) new age in the local optimum. Moreover, this other search point may be older than the search point that first entered the local optimum since this search point was assigned age 0 when it was created (as it was an improvement) and is thus younger than the other search points. If it is older, in the age-based variant again this age is used and introduced as (potentially) new age. In the optimistic value-based variant, however, the age of the better search points, i. e., the local optimum, is used and therefore the number of ages in the local optimum cannot increase. This does not only limit the number of different ages in the local optimum (in expectation it is $O(\mu \log(\mu)/n)$) but also ensures that the first point that entered the local optimum has maximal age in the local optimum.

This yields a lower bound on the number of steps before a partial restart occurs that we can exploit to prove a better upper bound on the expected optimization time.

The following lemma makes a strong assumption on the age distribution at the local optimum. Given this assumption we can prove a high probability for finding a global optimum. We discuss afterwards how these assumptions can be met.

Lemma 6 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using an arbitrary strategy for static pure aging from Definition 2 and an arbitrary strategy for selection for replacement from Definition 3, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Let the algorithm be completely at the local optimum, i. e., $\{f(x)\colon x \in C\} = \{(5/4)n\}$. Let $r+1$ denote the number of different ages in C. For each of the $r+1$ different ages, let the number of $x \in C$ with each age be $\Theta(\mu/r)$ for the subsequent τ steps or until a global optimum is found.

The probability that within the subsequent τ steps a global optimum is found is $\Omega(1)$.

Proof Due to our assumptions there are either r partial restarts in the subsequent τ steps or the global optimum is found. For each of these restarts we have probability

$$\Theta\Big(\frac{\mu/(r+1)}{\mu} \cdot \frac{\mu - \mu/(r+1)}{\mu}\Big) = \Theta\Big(\frac{1}{r+1} \cdot \Big(1 - \frac{1}{r+1}\Big)\Big) = \Theta\Big(\frac{1}{r}\Big)$$

to select one locally optimal search point and one search point generated uniformly at random by the partial restart for crossover. A crossover of these points creates a global optimum with probability $\Omega(1)$ (Lemma 1) so that each of the r partial restarts has success probability $\Omega(1)$. The probability to have at least one of these successful is

$$1 - \Big(1 - \Omega\Big(\frac{1}{r}\Big)\Big)^r = \Omega(1)$$

as claimed. □

Note that Lemma 6 holds for all variants of Algorithm 1. However, the assumption to always have $\Theta(\mu/r)$ search points for each of the r different ages is not realistic for all variants. However, for optimistic value-based aging in combination with the most frequent replacement strategy it is as the following lemma shows.

Lemma 7 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using optimistic value-based static pure aging from Definition 2 and the most frequent replacement strategy from Definition 3, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Consider the point of time when the number of different function values is reduced to 1 and all search points are in the local optimum, i. e., $\{f(x)\colon x \in C\} = (5/4)n$.

The conditions of Lemma 6 are met with probability $\Omega(1)$.

Proof Consider the first search point x to enter the local optimum. It is assigned age 0 at this point of time and as argued above no search point with a larger age can enter the local optimum. Thus, the first (partial) restart after x entered the local optimum occurs

after $\tau = \omega(\mu n \log \mu)$ steps. The expected takeover time for the complete population is $\Theta(\mu \log \mu)$. Thus, on expectation after $O(\mu \log \mu)$ steps we have the complete collection of search points in the local optimum and this can only change when the first partial restart occurs, i. e., after $\tau - O(\mu \log \mu) = \omega(\mu n \log \mu)$ steps. Thus, there is sufficient time to obtain a balanced distribution of the ages within the local optimum. □

Lemma 7 proves $q' = \Omega(1)$. All we need to obtain a better upper bound on the expected optimization time is a bound on p'. We recall that we already have such a bound.

Theorem 3 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using optimistic value-based static pure aging from Definition 2 and the most frequent replacement strategy from Definition 3, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Then, $E\left(T_{A,f}\right) = \Theta\Big(\Big(1 + \frac{n}{\mu \log \mu}\Big) \cdot \left(\tau + n^2 + \mu n \log n\right)\Big)$.

Proof We apply Corollary 3. According to Lemma 6 and Lemma 7 we have $q' = \Omega(1)$. Moreover, Lemma 5 yields $p' = \Omega(\mu \log(\mu)/n)$ for not too large μ and $p' = \Omega(1)$ otherwise. Together this yields the claimed upper bound. The lower bound is already contained in Corollary 2. □

Unfortunately, we are not able to prove a similar result for the other two variants of static pure aging. Since older search points can enter the local optimum we have no lower bound on the number of steps until a partial restart occurs. This implies that we cannot prove that the age structure is balanced and thus are unable to prove that the conditions of Lemma 6 are met. The improvement of the upper bounds to obtain tight bounds for these static pure aging variants is an open problem.

3.5 Fewest Replacement Strategy

In contrast to the former two replacement strategies, fewest replacement does not incorporate any age diversity mechanism. Even worse, diversity with respect to age is intentionally destroyed. In the next theorem we show that such an algorithm is with probability converging to 1 exponentially fast not able to optimize our considered example function.

Theorem 4 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using fewest replacement from Definition 3 and optimistic value-based aging from Definition 2, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and a maximal age $\tau = \omega(\mu n \log \mu)$.*

Then, the probability that Algorithm A does not find the optimum of f within 2^{cn} steps ($c > 0$ a sufficiently small constant) is bounded below by $1 - 2^{-\Omega(n)}$.

Proof Consider the first point in time when all search points are locally optimal. As seen in the proof of Lemma 7 the age of all search points is bounded above by $O(\mu \log \mu)$ at this point. With probability $1 - 2^{-\Omega(n)}$ all ages are bounded by $O(\mu n \log \mu)$. Thus,

there are still $\tau - O(\mu n \log \mu) = \omega(\mu n \log \mu)$ steps left before the first (partial) restart can occur.

Let x be a search point with some age $x.\text{age}$ that occurs most frequently in the current collection of search points. If there exist more than one such age we select an arbitrary one. Due to the replacement strategy the number of search points with that age will not decrease during the ongoing search process. Moreover, any time such a point is selected and copied the number of search points with the same age increases and the number of search points with other ages decreases. Thus, after expected $\Theta(\mu \log \mu)$, say $d\mu \log \mu$ ($d > 0$ constant), steps the collection of search points only consists of search points with only one single age, keeping the algorithm from performing a partial restart and yielding inefficient optimization time. Due to Markov's inequality the probability not to have such an event within $2d\mu \log \mu$ is at most $1/2$. Moreover, the probability not to have such an event in n rounds of $2d\mu \log \mu$ steps can be bounded above by $(1/2)^n$. Since $\tau = \omega(\mu n \log \mu)$ this yields the theorem. □

The proof of Theorem 4 does not work for age-based and pessimistic value-based static pure aging. These two different aging variants allow that search points may enter the local optimum that are older than the first search point that entered the local optimum. Since the age of these search points may be much older it cannot be ruled out that these search point cause a partial restart rather quickly and thus lead to successful optimization. However, it seems to be highly unlikely that very old search points survive long enough for this event to happen. We therefore speculate that Theorem 4 can be generalized for the other two aging variants, too. This, however, is currently an open problem.

3.6 Random Replacement Strategy

Finally we consider the random replacement variant. Here again no diversity mechanism with respect to age is used but in contrast to the fewest replacement diversity is just not cared about. Thus, we simply replace a random search point with worst fitness value. This is equivalent to the standard replacement strategy in evolutionary algorithms where age is not used at all. We prove that this also leads to inefficient optimization time if used in combination with the optimistic value-based aging strategy. We speculate that the same holds for the other aging strategies and discuss difficulties in proving this after the proof of the following result. This results demonstrates that age diversity mechanisms are an important concept for effective aging operators.

Theorem 5 *Consider the function $f\colon \{0,1\}^n \to \mathbb{R}$ from Definition 4. Let A denote Algorithm 1 using random replacement from Definition 3 and the optimistic value-based static pure aging variant from Definition 2, a size of the collection of search points $\mu \in \mathbb{N}$ with $\mu \geq 2$ and $\mu = n^{O(1)}$, crossover probability p_c with $0 < \varepsilon \leq p_c \leq 1 - \varepsilon < 1$ for a positive constant ε, $k = O(1)$ crossover points in k-point crossover, and maximal age $\tau = \omega(\mu n \log \mu)$.*

Then, the probability that Algorithm A does not find the optimum of f within 2^{cn} steps is bounded below by $1 - 2^{-\Omega((n \log \log \mu)/\log \mu)}$.

Proof We aim at proving that with probability close to 1 no partial restart will occur. This immediately implies the result since such a partial restart is needed for efficient optimization of f.

We consider the situation when $\max\{f(x)\colon x \in C\}$ is increased to $(5/4)n$, i. e., a first point enters the local optimum. Since this point is an improvement it is assigned age 0. We consider the subsequent steps and are interested in the first point of time when $\min\{f(x)\colon x \in C\} = (5/4)n$ holds, i. e., the complete collection of search points is in the local optimum. We claim that at this point of time x (and its descendants) have maximal age in C. Consider another search point z that enters the local optimum. Clearly, z was created either involving crossover or by mutation only. If it was created by mutation of a search point that is not locally optimal z is an improvement and $\text{age}.z = 0 < \text{age}.x$ holds. If z is a clone of a local optimum it inherits its age. Thus, by means of mutation only no search point with larger age can be introduced into the local optimum. Now, consider the case where z is created by means of crossover. If both search points are not locally optimal again z is an improvement. If at least one search point is a local optimum than z inherits its age since we are using optimistic value-based aging. Thus, also crossover cannot introduce a search point with age larger than $\text{age}.x$. Thus, x has maximal age. We claim that we have $\text{age}.x = O\big(\mu \log^2 \mu\big)$ at this point of time with probability $1 - 2^{-\Omega(\log^2 \mu)}$. As noted before the process of taking over the collection of search points is similar to the coupon collector process yielding an expected duration of $O(\mu \log \mu)$. The bound $\mu^{-\Omega(\beta)}$ on the probability for taking $\Omega(\beta\mu \log \mu)$ steps [30] carries over, too. Setting $\beta = \log \mu$ we obtain the bound $2^{-\Omega(\log^2 \mu)}$ as claimed. Note that during this process in all steps the order of growth of the probability of inserting another point to the local optimum is bounded above by the probability of adding another copy of x to the local optimum. Consequently, also with probability $1 - 2^{-\Omega(\log^2 \mu)}$ we have $\Omega(\mu)$ copies of x in C. This implies that in each selection such a search point is selected with probability $\Omega(1)$.

We consider the subsequent steps and claim that after some number of steps (where the number of steps will be discussed afterwards) x will have taken over the complete collection of search points (and thus there is only one age present in C) with probability $1 - 2^{-\Omega(\mu)}$. Note that if this happens before a partial restart happens no partial restart can happen anymore. Let n_x denote the number of copies of x in the beginning. Remember that we have $n_x = \Omega(\mu)$ with probability close to 1. In one round this number n_x may remain unchanged or it may either by increased or decreased by exactly 1. We want to prove that it will be increased to μ with probability close to 1.

Since we consider the situation when the whole collection of search points is in the local optimum we have that all search points have equal fitness and thus the age of the new search points equals $\max\{\text{age}.x, \text{age}.y\}$. Moreover, crossover of any two parents can only yield another local optimum as result. The event 'n_x is increased' can happen with and without crossover. Without crossover it happens if one such search point is selected (probability n_x/μ), it is not changed by mutation (probability $(1-1/n)^n$), and none of the n_x search points is selected for replacement (probability $(\mu - n_x)/\mu$). This leads to a contribution of $(1-p_c)(n_x/\mu)(1-1/n)^n(\mu-n_x)/\mu$ to Prob (n_x is increased) by this case. With crossover it happens if at least one such search point is selected (probability $1 - ((\mu - n_x)/\mu)^2 = (n_x/\mu)(2 - n_x/\mu)$), the result of crossover is not changed by mutation (probability $(1 - 1/n)^n$), and none of the n_x search points is

selected for replacement (probability $(\mu - n_x)/\mu$). Together we obtain

$$\begin{aligned}&\text{Prob}\,(n_x \text{ is increased})\\ &= (1-p_c)\frac{n_x}{\mu}\left(1-\frac{1}{n}\right)^n\frac{\mu-n_x}{\mu} + p_c\frac{n_x}{\mu}\left(2-\frac{n_x}{\mu}\right)\left(1-\frac{1}{n}\right)^n\frac{\mu-n_x}{\mu}\\ &= \frac{n_x(\mu-n_x)}{\mu^2}\left(1-\frac{1}{n}\right)^n\left(1+p_c-p_c\frac{n_x}{\mu}\right).\end{aligned}$$

The event 'n_x is decreased' can also happen with and without crossover. Without crossover it happens if some other search point is selected (probability $(\mu - n_x)/\mu$), it is not changed by mutation (probability $(1-1/n)^n$), and one of the n_x search points is selected for replacement (probability n_x/μ). This leads to a contribution of $(1-p_c)((\mu-n_x)/\mu)(1-1/n)^n n_x/\mu$ to $\text{Prob}\,(n_x \text{ is decreased})$ by this case. With crossover it happens if no such search point is selected (probability $((\mu-n_x)/\mu)^2$), the result of crossover is not changed by mutation (probability $(1-1/n)^n$), and one of the n_x search points is selected for replacement (probability n_x/μ). Together we obtain

$$\begin{aligned}&\text{Prob}\,(n_x \text{ is decreased})\\ &= (1-p_c)\frac{\mu-n_x}{\mu}\left(1-\frac{1}{n}\right)^n\frac{n_x}{\mu} + p_c\left(\frac{\mu-n_x}{\mu}\right)^2\left(1-\frac{1}{n}\right)^n\frac{n_x}{\mu}\\ &= \frac{n_x(\mu-n_x)}{\mu^2}\left(1-\frac{1}{n}\right)^n\left(1-p_c+p_c\frac{\mu-n_x}{\mu}\right)\end{aligned}$$

hold. For the sake of comparison we consider

$$\frac{\text{Prob}\,(n_x \text{ is increased})}{\text{Prob}\,(n_x \text{ is decreased})} = \frac{1+p_c-p_c n_x/\mu}{1-p_c+p_c(\mu-n_x)/\mu} = 1+\frac{p_c}{1-p_c\cdot n_x/\mu} > 1+p_c$$

and see a clear tendency towards increasing n_x. For the analysis we consider a Markov chain $X_0, X_1, \ldots$ on the state space $\{0,1,\ldots,\mu\}$ with $\text{Prob}\,(X_{t+1}=0)=1$ for $X_t=0$, $\text{Prob}\,(X_{t+1}=\mu)=1$ for $X_t=\mu$, $\text{Prob}\,(X_{t+1}=X_t+1)=(1+p_c)/(2+p_c)$, and $\text{Prob}\,(X_{t+1}=X_t-1)=1/(2+p_c)$ in all other cases. Clearly, all transition probabilities not explicitly stated are 0. This Markov chain corresponds to the algorithm conditioned on the event that n_x changes and is pessimistic with respect to having $n_x=\mu$ at some point of time. Moreover, the Markov chain corresponds exactly to the situation in the gambler's ruin theorem [15]. Remember that we have $n_x=\Omega(\mu)$, say $n_x=c\mu$, initially. Thus, the probability not to have $n_x=\mu$ at some point of time is bounded above by

$$\frac{(1+p_c)^{c\mu}-1}{(1+p_c)^{\mu}-1} = \left(\frac{1}{1+p_c}\right)^{(1-c)\mu}\cdot\left(1-\frac{(1+p_c)^{(1-c)\mu}+1}{(1+p_c)^{\mu}-1}\right) = 2^{-\Omega(\mu)}.$$

The expected duration of the random process described by the Markov chain is $O(\mu)$. However, this is different from the duration of the random process in the algorithm since we considered the situation conditioned that n_x is changed. Thus, we need to take into account the probability to change n_x in one step. This probability is given by

$$\text{Prob}\,(n_x \text{ is increased}) + \text{Prob}\,(n_x \text{ is decreased}) = \Omega\left(\frac{n_x(\mu-n_x)}{\mu^2}\right)$$

and we see that it is particularly small when $n_x = O(1)$ or $\mu - n_x = O(1)$ holds. We improve the trivial bound $O\big(\mu^2\big)$ on the duration to $O\big(\big(\mu \log^2 \mu\big) / \log\log \mu\big)$ in the following way.

Consider the situation with $n_x = O(1)$ for $\Theta(\mu \log \mu)$ steps. Since the probability to increase n_x by 1 is bounded below by $\Omega(1/\mu)$ in this situation we have on expectation $n_x = \Omega(\log \mu)$ after these steps. Consider another round of $\Theta(\mu \log \mu)$ steps. Now the probability to increase n_x is bounded below by $\Omega(\log(\mu)/\mu)$ and we expect to have $n_x = \Omega\big(\log^2 \mu\big)$ at the end. In general, after r such rounds we expect $n_x = \Omega(\log^r \mu)$. Thus, after $\Theta(\log(\mu)/\log\log \mu)$ rounds we have $n_x = \Omega\Big(\log^{\log(\mu)/\log\log \mu} \mu\Big) = \Omega(\mu)$ in the end. For $\mu - n_x$ the situation is symmetric (but reversed in time). Thus, we have an expected length of $O\big(\big(\mu \log^2 \mu\big) / \log\log \mu\big)$ as claimed.

In summation we have that on expectation after $O\big(\big(\mu \log^2 \mu\big) / \log\log \mu\big)$ steps the complete collection of search points is of the same age so that a partial restart is impossible. To obtain the result with probability very close to 1 we consider $\Theta(\mu n \log \mu)$ steps in total. These steps can be considered as $\Theta((n \log\log \mu)/\log \mu)$ repetitions of length $\Theta\big(\big(\mu \log^2 \mu\big) / \log\log \mu\big)$ each. This yields the desired bound on the probability. □

We speculate that a similar result holds for age-based and pessimistic value-based aging. However, proving this is an open problem. The difficulty is essentially the same as for Theorem 4.

4 Experimental supplements

The results presented in the preceding sections give new insights into what aging can achieve in randomized search heuristics. Our theoretical analyses give a coarse picture of the effects of aging, in particular with respect to partial restarts. Nevertheless, not all questions are answered. First of all, most of the derived bounds are not tight. Second, asymptotic results may not describe the situation for typical problem dimensions, in particular small problem sizes. As the size of the gap between upper and lower bound depends on the size of the collection of search points μ we further investigate the influence of this parameter in order to supplement our theoretical results. We hope that experiments give insights into possible improvements of either the lower or the upper bounds.

It is not obvious what good values for μ are. However, the theoretical results give hints. The bounds for the pessimistic value-based variant and the age-based variant indicate that a smaller size of the collection of search points leads to a smaller optimization time. For the optimistic value-based variant, the bounds suggest $\mu = \Theta(n/\log n)$ as a good choice. However, as our theoretical bounds are not tight, these speculations may be wrong.

We do all experiments with sizes $\mu \in \{2, \lfloor\sqrt{n}\rfloor, \lfloor n/\log n\rfloor, n\}$ for the collection of search points. Clearly, $\mu = 2$ is interesting as it is the smallest possible size and possibly a good choice for the pessimistic value-based and the age-based variant. For the same reason we pick $\mu = \lfloor n/\log n\rfloor$ for the optimistic value-based variant. The choice $\mu \approx \sqrt{n}$ has often turned out to be a good choice [18]. Moreover we are interested in the effects of sizes that are not sub-linear. Thus, we also pick $\mu = n$.

We require $\tau = \omega(\mu n \log \mu)$ and choose $\tau = \lfloor 6\mu n \log(\mu) \log n\rfloor$ for our experiments where the factor 6 helps for small values of n. All bounds work for arbitrary constant

	age-, pessimistic value-based	optimistic value-based
$\mu = 2$	$\Theta(n^2)$	$\Theta(n^3)$
$\mu = \lfloor\sqrt{n}\rfloor$	$\Omega(n^2)$, $O(n^{5/2})$	$\Theta(n^{5/2}/\log n)$
$\mu = \lfloor n/\log n\rfloor$	$\Omega(n^2 \log n)$, $O(n^3)$	$\Theta(n^2 \log n)$
$\mu = n$	$\Omega(n^2 \log^2 n)$, $O(n^3 \log^2 n)$	$\Theta(n^2 \log^2 n)$

Table 1 Bounds on the expected optimization time for example sizes of the collection of search points using the most frequent replacement strategy.

crossover probabilities p_c. We use $p_c = 0.5$, a medium sized value. All proofs work for any constant number k of crossover points. We consider the commonly used 1-point crossover.

We perform two sets of experiments. First of all, we consider the optimization times for the most frequent replacement and all static pure aging variants. Note, that in [25, 26] the smallest age distance replacement was analyzed in very much the same way. Second, we compare the optimization times of the other replacement strategies with most frequent replacement. The results of the different experiments are given in the following subsections.

4.1 Optimization Times of the Most Frequent Replacement Strategy

We analyze the optimization times of most frequent replacement combined with the three static pure aging strategies and different values for the size μ of the collection of search points. Table 1 shows the resulting bounds on the expected optimization times for this setting due to Corollary 2 for the lower bounds, Corollary 1 for the upper bound of age-based and pessimistic value-based aging and Theorem 3 for the improved upper bound of optimistic value-based aging. Note, that we also inserted the concrete value for τ that is used within the experiments when deriving these bounds. We see that for the considered parameter settings we have a gap of $\Theta(\mu)$ for age-based and pessimistic value-based aging while for the optimistic value-based we have a tight result.

For each setting we perform 100 independent runs and plot the results using box-and-whisker plots providing the mean together with the minimum, maximum, upper and lower quartile of the 100 runs for $n \in \{20, 40, \ldots, 1000\}$. Due to the excessive computation time, we consider only $n \in \{20, 40, \ldots, 340\}$ for $\mu = n$ (all variants) and $n \in \{20, 40, \ldots, 460\}$ for $\mu = 2$ (optimistic value-based). The results are shown in Figure 3 for age-based aging, in Figure 4 for optimistic value-based aging and in Figure 5 for pessimistic value-based aging where the number of iterations are drawn in logarithmic scale. To facilitate comparison we plot the medians for all sizes of the collection of search points in one joint diagram in each case (Figure 3-5, bottom) in linear scale.

Like in [25,26] for smallest age distance replacement, it is obvious that also for most frequent replacement the variance decreases with increasing size of the collection of search points. This is due to the fact that the probability for a partial restart increases with increasing μ. Consider the situation just after the first individual reached the local optimum. In the extreme case $\mu = 2$ there is only one other individual left that needs to enter the local optimum with a different age in order to allow for a partial restart. For larger μ more trials are possible. If the algorithm fails to perform a partial restart, a complete restart is required. Certainly, complete restarts are rather expensive and lead to larger variances in the optimization time. For $\mu = n$ we see that generally no

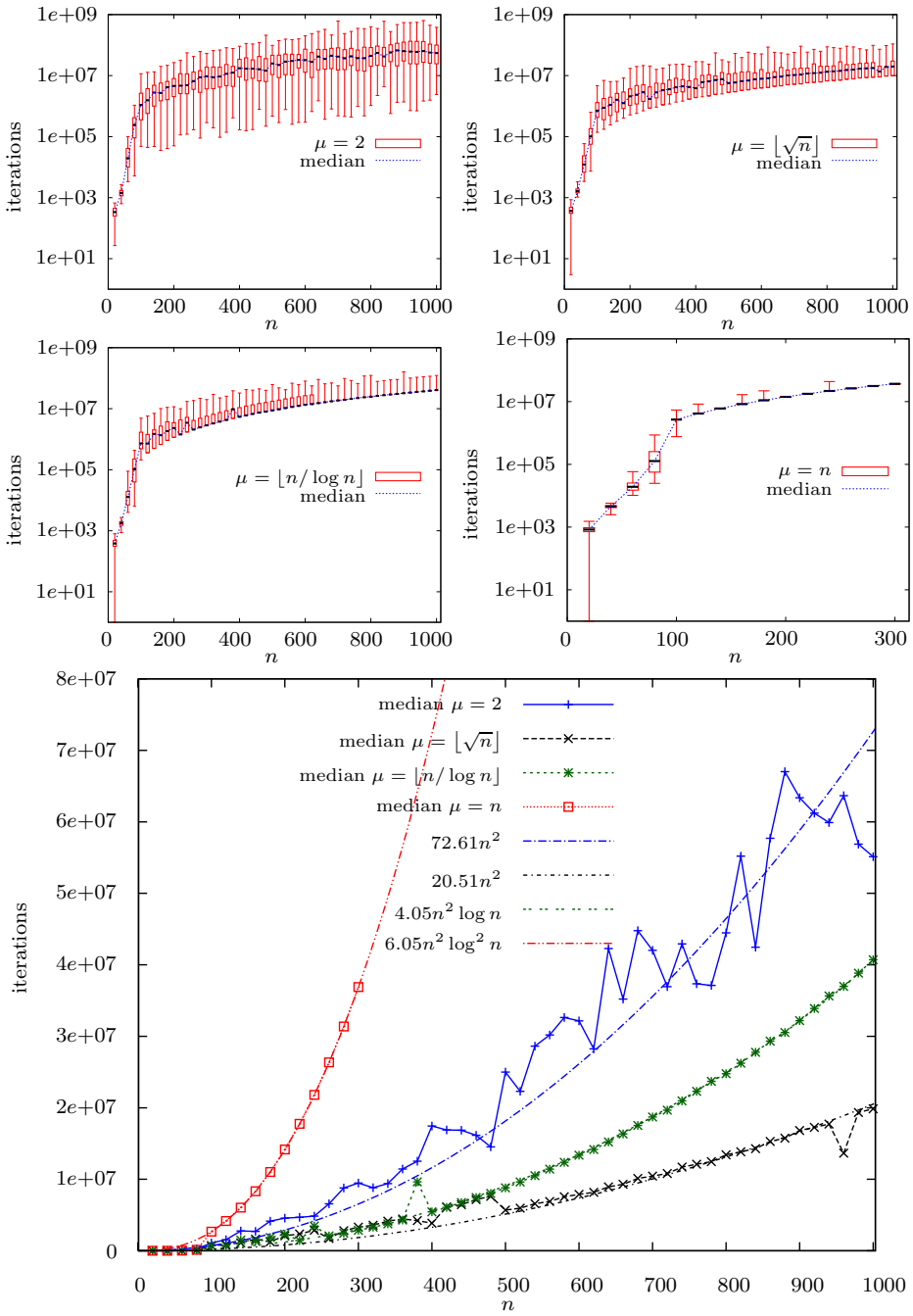


Fig. 3 Experimental results for most frequent replacement and age-based aging with different values for the size μ of the collection of search points.

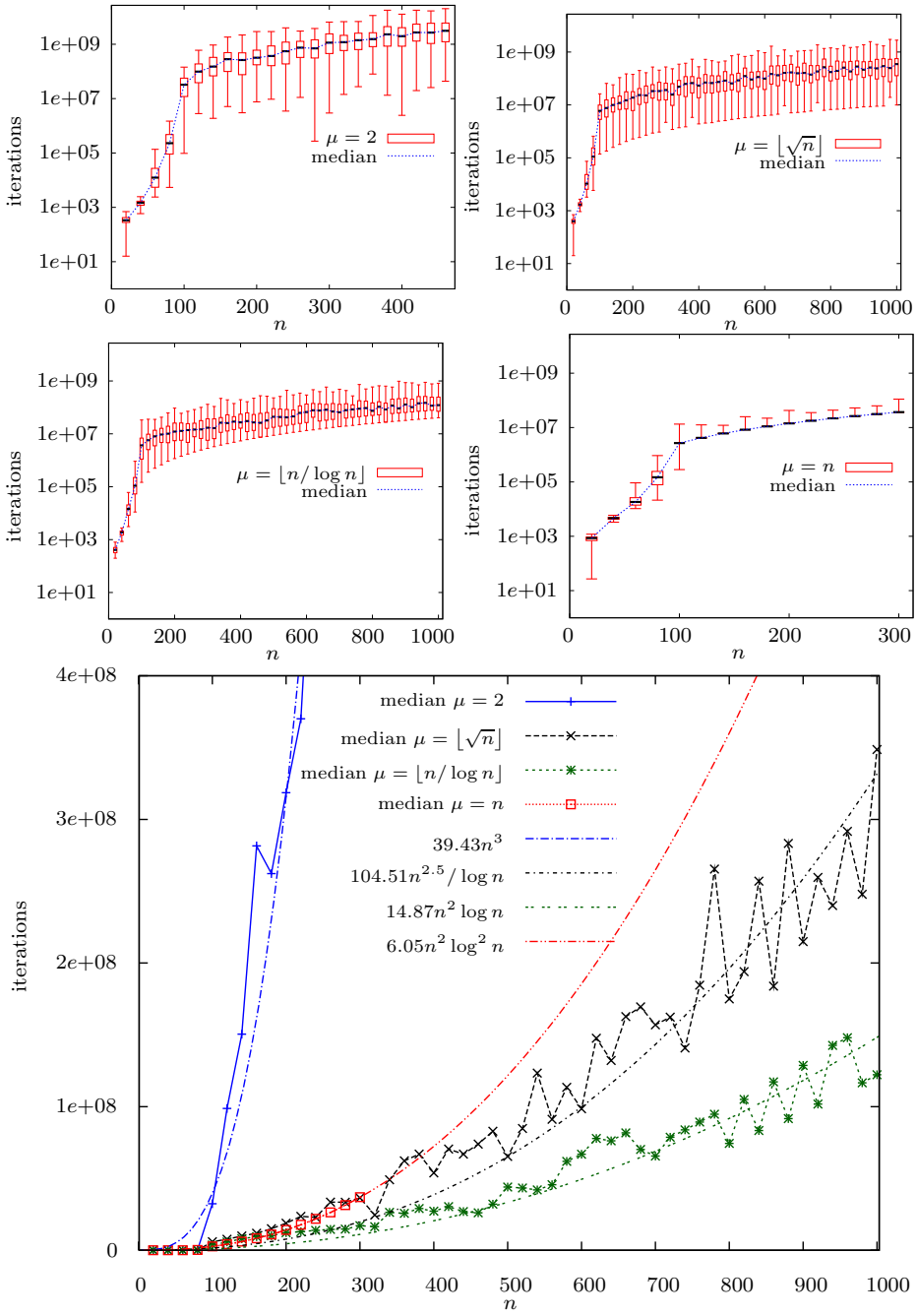


Fig. 4 Experimental results for most frequent replacement and optimistic value-based aging with different values for the size μ of the collection of search points.

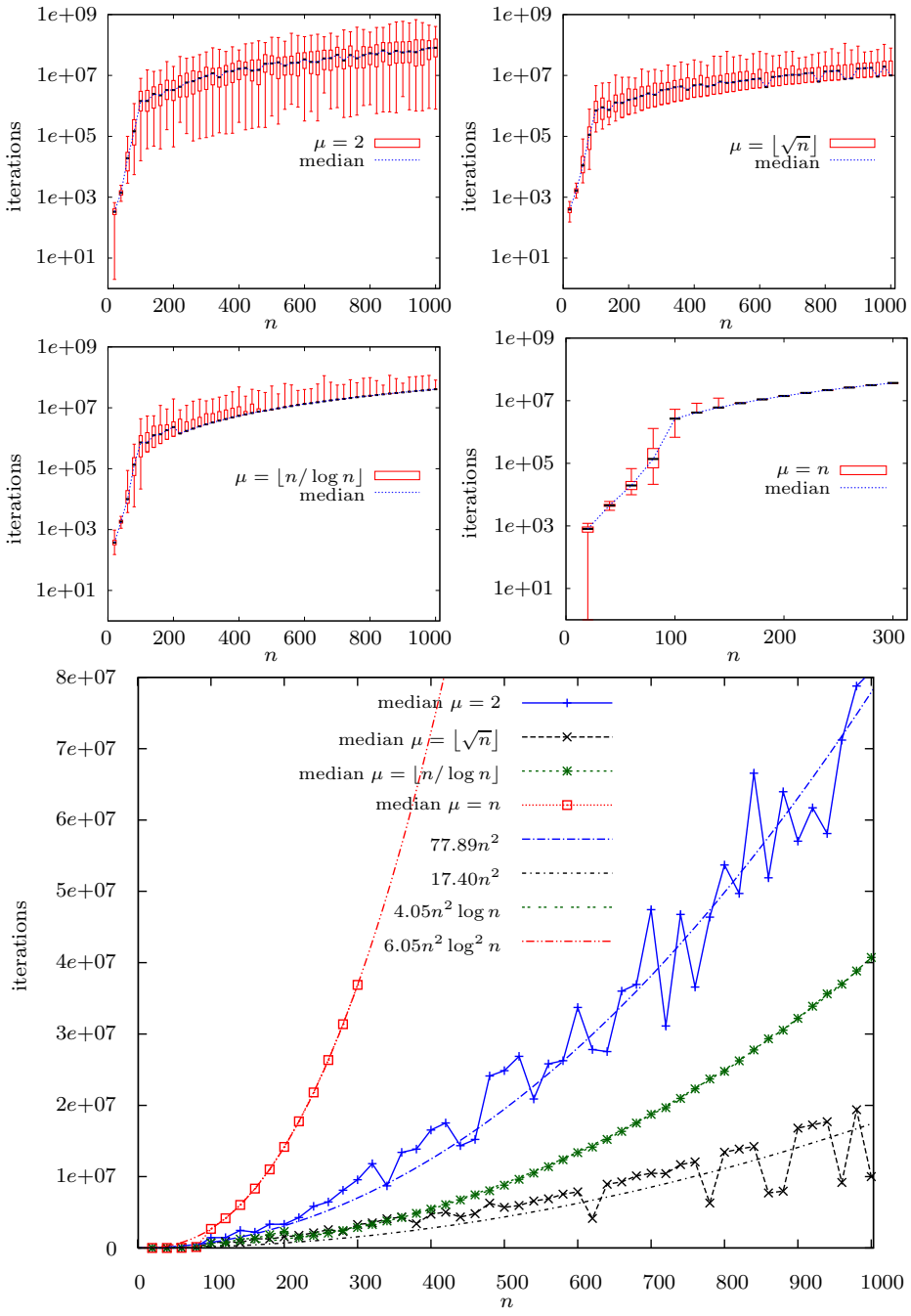


Fig. 5 Experimental results for most frequent replacement and pessimistic value-based aging with different values for the size μ of the collection of search points.

complete restarts occur during the optimization process and we succeed in performing a suitable crossover operation when we reach the local optimum for the first time.

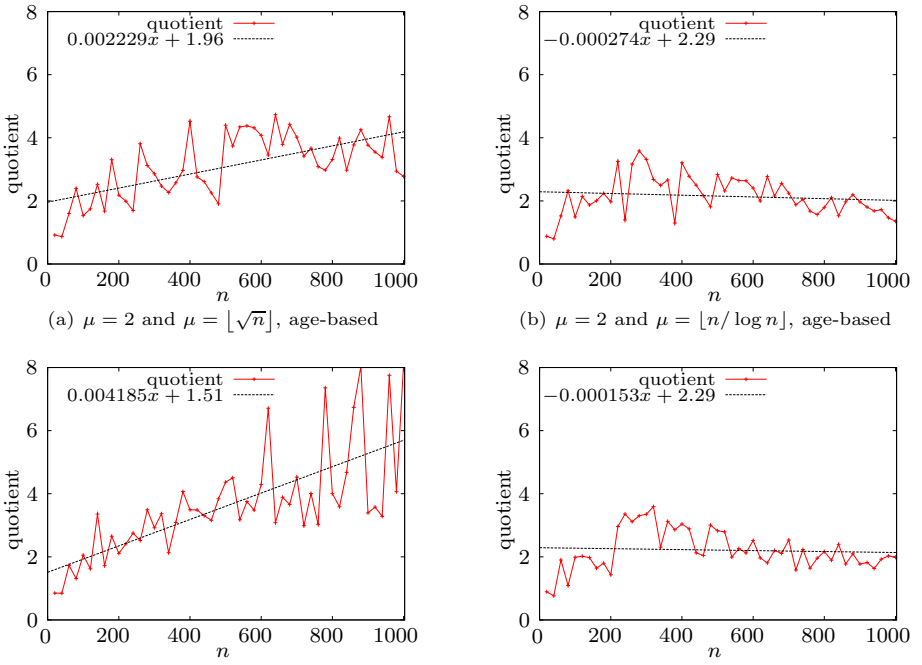
Moreover, we see that the variance is larger for optimistic value-based aging than for the other two variants. This is due to the fact, that here crossover is not able to create a search point with a different age in the local optimum whereas this is possible for the other two aging variants. Additionally, the variance for age-based aging is slightly larger than for pessimistic value-based aging. This can again be explained with the effects of the crossover operator: in pessimistic value-based aging always the age of the worse search point is inherited (if it is not an improvement). In case of a crossover of a locally optimal search point x and a worse search point y that creates another locally optimal search point z, this means that another age is introduced if y.age was not already present in the local optimum. Since the first locally optimal search point is always created by means of mutation, this is always the case as long as there is only one age value in the local optimum. Thus, in this situation a suitable crossover operation always introduces a second age. This is not true for the age-based variant as additionally $y.\text{age} > x.\text{age}$ must hold. We conclude that pessimistic value-based aging is more robust than age-based aging with respect to introducing a second age and hence allowing for a partial restart whereas optimistic value-based aging is least robust in this respect.

We compare the effects of the size of the collection of search points (Figure 3-5, bottom). First, note that not only the theoretical upper and lower bounds for age-based and pessimistic value-based aging are identical but in fact there is hardly any difference visible in the experimental results. This lack of empirical difference is also present in the experimental results for $\mu = n$ in all three aging variants. For the optimistic value-based variant, we see that the experimental results are in good accordance with the theoretical results, namely $\mu = \lfloor n/\log n \rfloor$ being the fastest and $\mu = 2$ being by far the worst.

For the age-based and pessimistic value-based variant, surprisingly, the algorithm with size $\mu = 2$ is clearly outperformed by its counterparts with sizes $\mu = \lfloor \sqrt{n} \rfloor$ and $\mu = \lfloor n/\log n \rfloor$. This shows that (at least for not too large values of n) the asymptotic theoretical results are misleading from a practical point of view. This is due to the large constants hidden in the derived asymptotic bounds and thus, these bounds do not reflect the actual optimization times on the small input sizes considered in the experimental study. However, it is not clear how large n needs to be chosen in order to obtain results that are in correspondence with the asymptotic theoretical results.

We speculate that our upper bounds are not tight. We support this hypothesis by plotting the fitted lower bounds together with the empirical mean in Figure 3-5 (bottom) and find a good fit in all cases. That larger sizes of the collection of search points outperform the choice $\mu = 2$ at least partially contradicts the theoretical results. Thus, we take a closer look by comparing the quotients of the observed means.

The theoretical bounds predict the quotient for $\mu = 2$ over $\mu = \lfloor n/\log n \rfloor$ to converge to 0. For $\mu = 2$ over $\mu = \lfloor \sqrt{n} \rfloor$ it should be bounded above by a positive constant. Note that the theoretical bounds are asymptotic and predict this behavior for $n \to \infty$. For $\mu = 2$ over $\mu = \lfloor n/\log n \rfloor$ (Figure 6(b) and Figure 6(d)) we see that after an increase for small values of n the quotient does indeed decrease. We fit the graph of the linear function $a \cdot x + b$ to the data and see that already for $n \leq 1000$ the results match the asymptotic bounds. Things are different for $\mu = 2$ over $\mu = \lfloor \sqrt{n} \rfloor$ (Figure 6(a) and Figure 6(c)). Instead of being obviously bounded the quotient increases. This impression is confirmed when fitting $a \cdot x + b$ to the data. It



(a) $\mu = 2$ and $\mu = \lfloor \sqrt{n} \rfloor$, age-based

(b) $\mu = 2$ and $\mu = \lfloor n/\log n \rfloor$, age-based

(c) $\mu = 2$ and $\mu = \lfloor \sqrt{n} \rfloor$, pessimistic value-based

(d) $\mu = 2$ and $\mu = \lfloor n/\log n \rfloor$, pessimistic value-based

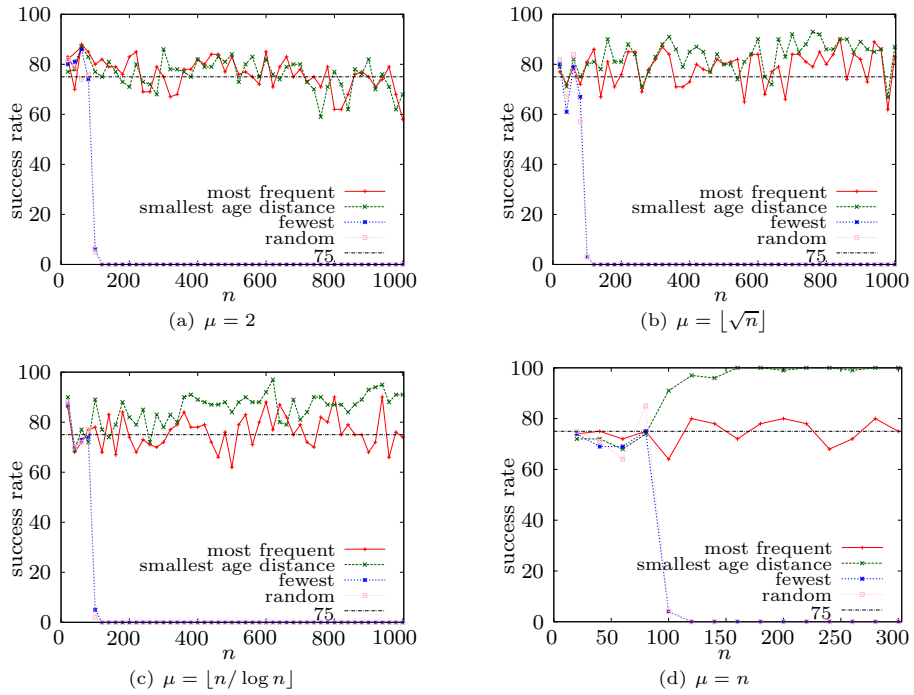
Fig. 6 Quotient of the observed medians.

is impossible to say from these experiments if the values of n considered are still too small or if the high variance is to blame.

4.2 Comparison of the Different Replacement Strategies

In order to compare the four replacement strategies from Definition 3 we perform experiments with all four of them and the parameter settings from above. We fix the maximal number of iterations executed to the corresponding upper quartile from the first set of experiments. The results are given as a plot for each μ and aging variant showing the number of successful runs within 100 independent runs for considered values of n: Figure 7 for age-based aging, Figure 8 for optimistic value-based aging and Figure 9 for pessimistic value-based aging.

In all settings considered it becomes apparent that the success rate of random replacement and fewest replacement starts decreasing for $n \approx 100$ and then converges to 0 very quickly. We can conclude that already for quite small values of n these two strategies are ineffective since with high probability one single age takes over the population in the local optimum, preventing the algorithm from performing a partial restart. This can be observed for all three aging variants. Note, that we only derived bounds for these two replacement strategies in the optimistic value-based variant. However, the experimental results support our speculation that similar bounds hold for the other two aging strategies.



(a) $\mu = 2$ (b) $\mu = \lfloor \sqrt{n} \rfloor$

(c) $\mu = \lfloor n / \log n \rfloor$ (d) $\mu = n$

Fig. 7 Experimental results for the different replacement strategies and age-based aging.

In contrast to that observation both most frequent replacement and smallest age distance replacement are effective. Note, that for most frequent replacement we expect a success rate of 75% since we use the upper quartile of the number of iterations during 100 independent runs of this algorithmic variant. Our expectations are met in all settings considered here. Surprisingly, the success rate of smallest age distance replacement is larger for increasing size of the collection of search points μ for the limited range of values inspected. This is in particular true for age-based and pessimistic value-based aging where for $\mu = n$ a success rate of nearly 100% is realized. Note, that the effect is already visible for $\mu = \lfloor \sqrt{n} \rfloor$ and $\mu = \lfloor n / \log n \rfloor$. For optimistic value-based aging the superiority of smallest age distance replacement is not that obvious. It appears that the success rate for $\mu = \lfloor \sqrt{n} \rfloor$ and $\mu = \lfloor n / \log n \rfloor$ is slightly higher but surprisingly it is not for $\mu = n$. Thus, it is not clear if these effects are only due to the random fluctuation.

The observations can be explained as follows. Since in optimistic value-based aging crossover does not help in creating different ages in the local optimum, we expect less different ages in the local optimum in comparison to the other two aging variants. Additionally, most frequent replacement tends to quickly equally distribute the quantities of the different age values. Having more age values, decreases the proportion of a single age value and decreases the probability to perform an appropriate crossover operation in a single restart. In smallest age distance replacement, the initial proportions of the different ages are not changed. It seems that for age-based and pessimistic value-based aging partial restarts and the effects of several successive partial restarts are more ef-

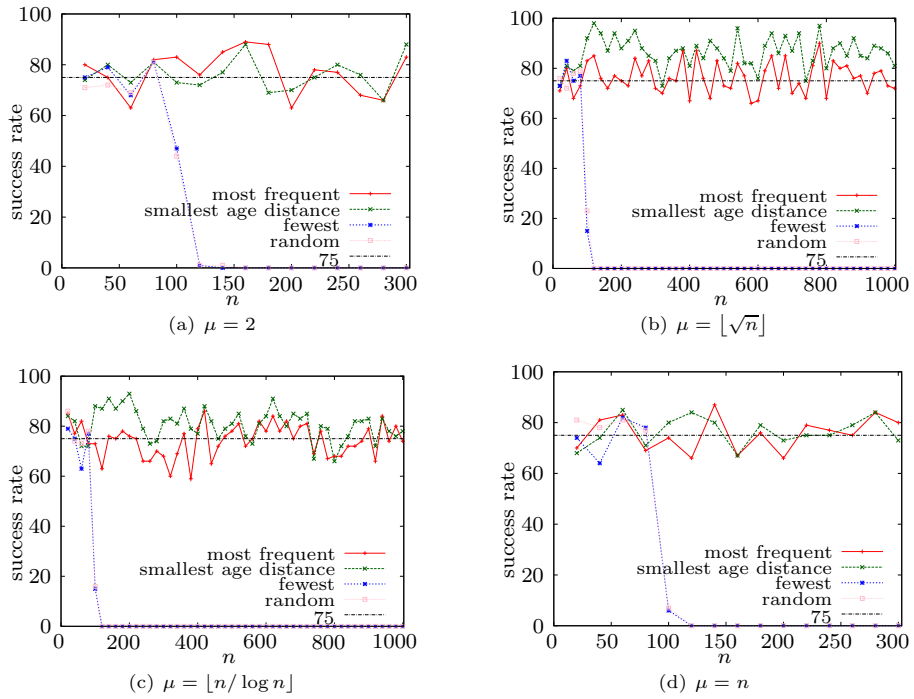


Fig. 8 Experimental results for the different replacement strategies and optimistic value-based aging.

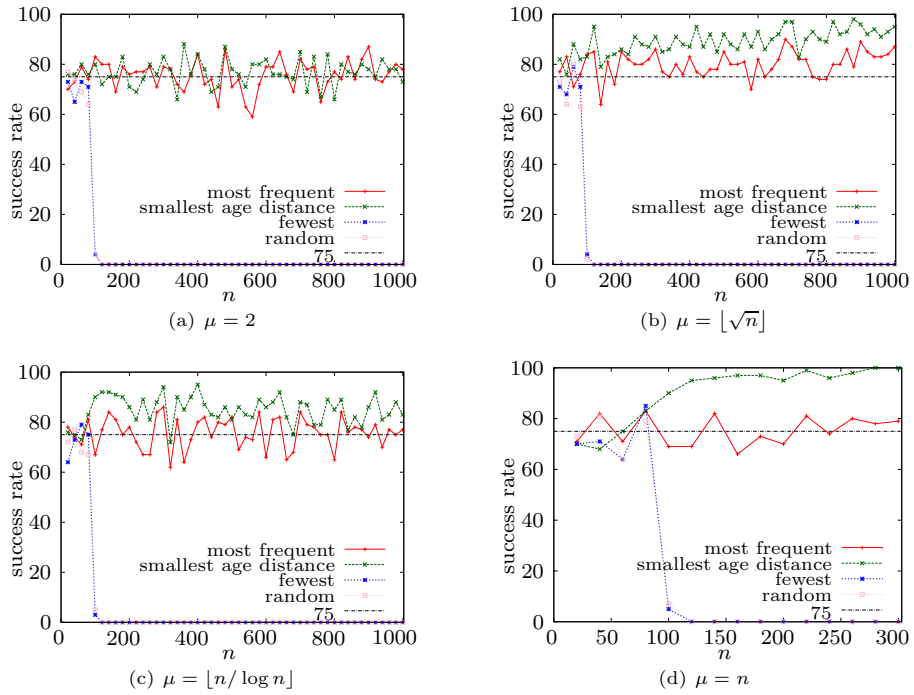
fective if the quantities of different age values are not equally distributed whereas for optimistic value-based aging both mechanisms yield very similar performance.

5 Conclusions

Aging is a nature-inspired mechanism that aims at maintaining some degree of diversity within a collection of search points. It has been implemented and applied in different general randomized search heuristics, most notably in artificial immune systems and evolutionary algorithms. In the context of artificial immune systems, static pure aging is the most important aging mechanism.

In static pure aging search points are rewarded for being an improvement by assigning them age 0. This lets them explore the search space for the complete maximal lifespan τ. Search points that fail to excel over the search points they originate from are punished by inheriting their age. While this general idea is clear, there are many different concrete ways to implement it. We have investigated twelve different concrete instantiations of static pure aging and analyzed their performance on one carefully chosen example problem. The example problem is constructed in a way that the use of aging is necessary to allow for efficient optimization. Theoretical and empirical analyses allow for a more complete understanding of static pure aging.

The main finding is that static pure aging can be sub-divided into the aging strategy and a replacement strategy. The aging strategy determines the way a new search point is assigned its age. This is not entirely clear in the case the new search point fails to be



(a) $\mu = 2$ (b) $\mu = \lfloor \sqrt{n} \rfloor$

(c) $\mu = \lfloor n / \log n \rfloor$ (d) $\mu = n$

Fig. 9 Experimental results for the different replacement strategies and pessimistic value-based aging.

an improvement. The replacement strategy determines the way the new search point is introduced in the collection of search points. Both aspects can be implemented in different ways and arbitrarily combined with each other. We considered three aging strategies and four replacement strategies yielding twelve concrete variants of static pure aging. These three main parameters of aging, the aging strategy, the replacement strategy, and the maximal age allow for a rich, diverse, and interesting behavior of static pure aging. We presented an in-depth analysis using theoretical analysis as well as experimental investigation.

We summarize the results of the theoretical analysis in Table 2. We considered twelve variants of static pure aging that result as a combination of three different aging strategies with four different replacement strategies.

For the fewest and random replacement strategies we proved that optimistic value-based aging leads to inefficient optimization of the example function. In other words, actively destroying age diversity renders static pure aging in the optimistic value-based variant useless. We conjecture that this holds for the other two aging variants, too. Finding such a proof is an open problem. The other two replacement strategies both aim at generating and preserving some age diversity and they both lead to efficient optimization of the example function for each aging strategy. The optimistic value-based variant restricts the role crossover can play when optimizing the example function. This facilitates the analysis and allows us to prove asymptotically tight bounds for the most frequent replacement strategy. Finding tight bounds for the other five efficient variants is still an open problem.

Table 2 Summary of the bounds on the expected optimization time for all twelve variants of Algorithm 1.

	age-based	**pessimistic value-based**	**optimistic value-based**
most frequent	$O\big(\mu \cdot (\tau + n^2 + \mu n \log n)\big)$ Corollary 1 $\Omega\big(\tau + n^2 + \mu n \log n\big)$ Corollary 2	$O\big(\mu \cdot (\tau + n^2 + \mu n \log n)\big)$ Corollary 1 $\Omega\big(\tau + n^2 + \mu n \log n\big)$ Corollary 2	$\Theta\Big(\Big(1 + \frac{n}{\mu \log \mu}\Big) \cdot (\tau + n^2 + \mu n \log n)\Big)$ Theorem 3
smallest age distance	$O\big(\mu \cdot (\tau + n^2 + \mu n \log n)\big)$ Theorem 1 $\Omega\big(\tau + n^2 + \mu n \log n\big)$ Theorem 2	$O\big(\mu \cdot (\tau + n^2 + \mu n \log n)\big)$ Theorem 1 $\Omega\big(\tau + n^2 + \mu n \log n\big)$ Theorem 2	$O\Big(\Big(\mu + \frac{n}{\log \mu}\Big) \cdot (\tau + n^2 + \mu n \log n)\Big)$ Theorem 1 $\Omega\Big(\Big(1 + \frac{n}{\mu \log \mu}\Big) \cdot (\tau + n^2 + \mu n \log n)\Big)$ Theorem 2
random	$2^{O(n)}$	$2^{O(n)}$	$2^{\Theta(n)}$ Theorem 5
fewest	$2^{O(n)}$	$2^{O(n)}$	$2^{\Theta(n)}$ Theorem 4

The supplementary experimental study gives further insights into the algorithms under consideration. On one hand, the role of the size of the collection of search points for most frequent replacement is investigated. We see that just like in smallest age distance replacement analyzed in [25,26], the variance is mostly dominated by the number of restarts needed and this number decreases with increasing population size. Thus, larger populations come with the benefit of greater reliability but at the cost of increased optimization time if the population size becomes too large. For small and reasonable problem sizes and rather small population sizes the theoretical bounds are misleading. Increasing the population size under these circumstances actually increases efficiency.

On the other hand, the different replacement strategies are compared. It becomes obvious that fewest replacement as well as random replacement lead to inefficient optimization even for small problem sizes. Moreover, for optimistic value-based aging most frequent and smallest age distance replacement behave very similar. However, this is not true for the other two aging variants. Here, it appears that smallest age distance replacement is slightly more efficient than most frequent replacement. It is an open problem to determine if the empirical differences actually corresponds to asymptotical different bounds.

The combination of empirical and theoretical results together shed light on the functioning of aging. It has become evident that the structure of aging involves not only an important role for the maximal age but also for the specific aging and replacement strategies. These aspects can all be studied in isolation and combined in almost arbitrary ways. By means of the specific example function it has become evident that aging can achieve beneficial effects that are difficult if not impossible to achieve otherwise. These effects, however, are very sensitive with respect to the implementation detail. It is therefore for any paper concerned with aging, theoretical as well as practical, a necessity to report every detail of the specific kind of aging involved. Only this can lead to joint research that paves the way for a more informed and efficient use of aging for solving important problems.

Acknowledgements The authors thank Nicola Beume for suggesting to consider different replacement strategies. This material is based in part upon works supported by the Science Foundation Ireland under Grant No. 07/SK/I1205.

References

1. D. A. Ackley. *A Connectionist Machine for Genetic Hillclimbing*. Kluwer Academic Publishers, 1987.
2. M. Castrogiovanni, G. Nicosia, and R. Rascunà. Experimental analysis of the aging operator for static and dynamic optimisation problems. In B. Apolloni, R. J. Howlett, and L. C. Jain, editors, *Proceedings of the 11th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2007)*, volume 4694 of *Lecture Notes in Computer Science*, pages 804–811. Springer, 2007.
3. D.-H. Choi. Cooperative mutation based evolutionary programming for continuous function optimization. *Operations Research Letters*, 30(3):195–201, 2002.
4. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*, volume 2nd. MIT Press, 2001.
5. V. Cutello, D. Lee, S. Leone, G. Nicosia, and M. Pavone. Clonal selection algorithm with dynamic population size for bimodal search spaces. In L. Jiao, L. Wang, X. Gao, J. Liu, and F. Wu, editors, *Proceedings of the 2nd International Conference on Advances in Natural Computation (ICNC 2006)*, volume 4221 of *Lecture Notes in Computer Science*, pages 949–958. Springer, 2006.

6. V. Cutello, G. Morelli, G. Nicosia, and M. Pavone. Immune algorithms with aging operators for the string folding problem and the protein folding problem. In G. R. Raidl and J. Gottlieb, editors, *Proceedings of the 5th European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP 2005)*, volume 3448 of *Lecture Notes in Computer Science*, pages 80–90. Springer, 2005.
7. V. Cutello and G. Nicosia. Multiple learning using immune algorithms. In *Proceedings of the 4th International Conference on Recent Advances in Soft Computing (RASC 2002)*, pages 102–107, 2002.
8. V. Cutello, G. Nicosia, and M. Pavone. A hybrid immune algorithm with information gain for the graph coloring problem. In *Proceedings of the 5th Annual Conference on Genetic and Evolutionary Computation (GECCO 2003)*, volume 2723 of *Lecture Notes in Computer Science*, pages 171–182. Springer, 2003.
9. V. Cutello, G. Nicosia, and M. Pavone. Exploring the capability of immune algorithms: A characterization of hypermutation operators. In G. Nicosia, V. Cutello, P. J. Bentley, and J. Timmis, editors, *Proceedings of the 3rd International Conference on Artificial Immune Systems (ICARIS 2004)*, volume 3239 of *Lecture Notes in Computer Science*, pages 263–276. Springer, 2004.
10. V. Cutello, G. Nicosia, and M. Pavone. An immune algorithm with hyper-macromutations for the dill's 2d hydrophobic-hydrophilic model. In *Proceedings of the 6th IEEE Congress on Evolutionary Computation (CEC 2004)*, pages 1074–1080. IEEE Press, 2004.
11. V. Cutello, G. Nicosia, and M. Pavone. An immune algorithm with stochastic aging and Kullback entropy for the chromatic number problem. *Journal of Combinatorial Optimization*, 14(1):9–33, 2007.
12. V. Cutello, G. Nicosia, M. Pavone, and J. Timmis. An immune algorithm for protein structure prediction on lattice models. *IEEE Transactions on Evolutionary Computation*, 11(1):101–117, 2007.
13. V. Cutello, G. Nicosia, M. Romeo, and P. S. Oliveto. On the convergence of immune algorithms. In *Proceedings of the IEEE Symposium on Foundations of Computational Intelligence (FOCI 2007)*, pages 409–415. IEEE Press, 2007.
14. D. Dasgupta and L. F. Niño. *Immunological Computation: Theory and Applications*. Auerbach, 2008.
15. W. Feller. *An Introduction to Probability Theory and Its Applications. Volume I*. Wiley, 1968.
16. T. Friedrich, P. S. Oliveto, D. Sudholt, and C. Witt. Analysis of diversity-preserving mechanisms for global exploration. *Evolutionary Computation*, 17(4):455–476, 2009.
17. A. Ghosh, S. Tsutsui, and H. Tanaka. Individual aging in genetic algorithms. In *Australian New Zealand Conference on Intelligent Information Systems*, pages 276–279. IEEE Press, 1996.
18. G. Harik, E. Cantú-Paz, D. Goldberg, and B. Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–253, 1999.
19. G. S. Hornby. ALPS: the age-layered population structure for reducing the problem of premature convergence. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation (GECCO 2006)*, pages 815–822. ACM Press, 2006.
20. G. S. Hornby. Steady-state ALPS for real-valued problems. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation (GECCO 2009)*, pages 795–802. ACM, 2009.
21. G. S. Hornby. A steady-state version of the age-layered population structure EA. In D. E. Goldberg, J. R. Koza, R. Riolo, U.-M. O'Reilly, and T. McConaghy, editors, *Genetic Programming Theory and Practice VII*, Genetic and Evolutionary Computation, pages 87–102. Springer, 2010.
22. C. Horoba, T. Jansen, and C. Zarges. Maximal age in randomized search heuristics with aging. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation (GECCO 2009)*, pages 803–810. ACM Press, 2009.
23. T. Jansen and I. Wegener. On the analysis of evolutionary algorithms – a proof that crossover really can help. *Algorithmica*, 34(1):47–66, 2002.
24. T. Jansen and C. Zarges. Comparing different aging operators. In *Proceedings of the 8th International Conference on Artificial Immune Systems (ICARIS 2009)*, volume 5666 of *Lecture Notes in Computer Science*, pages 95–108. Springer, 2009.
25. T. Jansen and C. Zarges. Aging beyond restarts. In *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation (GECCO 2010)*, pages 705–712. ACM Press, 2010.

26. T. Jansen and C. Zarges. On the benefits of aging and the importance of details. In *Proceedings of the 9th International Conference on Artificial Immune Systems (ICARIS 2010)*, volume 6209 of *Lecture Notes in Computer Science*, pages 61–74. Springer, 2010.
27. K. A. D. Jong. *Evolutionary Computation. A Unified Approach.* MIT Press, 2006.
28. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.
29. N. Kubota and T. Fukuda. Genetic algorithms with age structure. *Soft Computing*, 1(4):155–161, 1997.
30. R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.
31. H.-P. Schwefel and G. Rudolph. Contemporary evolution strategies. In F. Morán, A. Moreno, J. J. M. Guervós, and P. Chacón, editors, *Proceedings of the 3rd European Conference on Artificial Life (ECAL 1995)*, volume 929 of *Lecture Notes in Computer Science*, pages 893–907. Springer, 1995.
32. G. Stracquadanio, C. Drago, V. Romano, and G. Nicosia. An immunological algorithm for doping profile optimization in semiconductors design. In *Proceedings of the 9th International Conference on Artificial Immune Systems (ICARIS 2010)*, volume 6209 of *Lecture Notes in Computer Science*, pages 213–222. Springer, 2010.
33. C. Witt. Runtime analysis of the (μ+1) EA on simple pseudo-Boolean functions. *Evolutionary Computation*, 14(1):65–86, 2006.